VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zvýšení zabezpečení VoIP serveru
Increasing the Security of VoIP Server

2014 **Ondřej Pinda**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Ondřej Pinda**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R059 Mobilní technologie

Téma: Zvýšení zabezpečení VoIP serveru
Increasing the Security of VoIP Server

Zásady pro vypracování:

Cílem práce je prozkoumat současné možnosti a používané postupy pro zabezpečení serveru používající jádro GNU/Linux. Teoretická část práce důkladně popisuje jednotlivé technologie spojené s ochranou serveru proti různým druhům útoků. Na teoretický rozbor navazuje praktická část, kdy je server zabezpečen podle poznatků z teoretické části. Zabezpečení serveru je následně otestováno s uvedeným shodnocením zabezpečení serveru.

Body zadání:
1. Teoretický úvod do problematiky zabezpečení serveru na bázi GNU/Linux, správa serveru pomocí protokolu SSH
2. Popis problematiky PKI a její využití na straně serveru
3. Nasazení firewallu a použití IDS/IPS systémů
4. Monitoring serveru a popis specifických požadavků VoIP serveru
5. Nasazení bezpečnostních opatření na základě provedených teoretických rozborů
6. Ověření funkcionality použitých bezpečnostních kroků a shodnocení dosažených výsledků

Seznam doporučené odborné literatury:

BAUER M. D., Linux Server Security, O'Reilly Media, 2005, ISBN: 0-596-00670-5
TURNBULL J., Hardening Linux, Apress, 2005 ISBN: 1-59059-444-4

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Jakub Šafařík**

Datum zadání: 16.11.2012
Datum odevzdání: 07.05.2013

VYSOKÁ ŠKOLA BÁŇSKÁ TECHNICKÁ UNIVERZITA * OSTRAVA * Fakulta elektrotechniky a informatiky

prof. RNDr. Vladimír Vašinek, CSc.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne 07. 05. 2014

podpis studenta

Poděkování

Rád bych poděkoval Ing. Jakubovi Šafaříkovi za odbornou pomoc a konzultaci při vytváření této bakalářské práce.

Abstrakt

Tato práce slouží k prezentaci současných možností zabezpečení VoIP serveru a jeho monitoringu. Čtenář je obeznámen s těmito třemi nástroji Snort, Suricata a Bro sloužících k odhalení útoku, které jsou volně ke stažení. K monitoringu daného serveru bude použito systému Zabbix a Nagios.

První polovina práce rozebírá jednotlivé části zabezpečení pouze teoreticky a v následujících kapitolách je praktická ukázka nasazení těchto systémů v praxi. Kde budou podrobně popsány jednotlivé kroky každé instalace i s následným nastavením konfiguračních souborů. Testování IDS/IPS bude probíhat na dvou počítačích. Na první stanici bude postupně instalován každý z výše jmenovaných nástrojů a ze stanice druhé budou generovány útoky zvané invite flood.

Druhá polovina praktické části zabývající se monitoringem bude zaměřena na základní nastavení nutné k monitoringu serveru. K realizaci budou opět použity dvě stanice. První stanice bude představovat náš server s nainstalovaným agentem a na stanici druhé samotný monitorovací systém.

Klíčová slova

VoIP, detekce průniku, ochrana proti průniku, monitoring serveru, IDS, IPS, zabezpečení, útok, Snort, Bro, Suricata, Zabbix, Nagios,

Abstract

This thesis presents actual options of security VOIP server and his monitoring. The reader is familiar with three tools named Snort, Suricata a Bro, which are used to detection of attack. These are free to download. For monitoring of specific server should be use system Zabbix and Nagios.

The individual chapters of security are analyzed in the first half of thesis in theory. The following part contains practical demonstration of application these systems in practice, where all steps of instalation and the following settings of configuration´s files are desribed. Testing of IDS/IPS must be pass on 2 computers, on the first computer there all named tools should be install and the second computer should be use to generation of attack named invite flood.

The second part of practical chapter is concentrating on basic setting, which is necessary for monitoring of server. For realization must be use two stations again. The first computer should be represent our server wiht installed agent and on the second station there should be monitoring system.

Key words

VoIP, Intrusion detection system, intrusion protection system,server monitoring, IDS, IPS,security, attack, Snort, Bro, Suricata, Zabbix, Nagios

Seznam použitých zkratek

Zkratka	Význam
ARP	Address Resolution Protocol
ASK	Asymmetric cryptography
PKI	Public Key infrastructure
CA	Certificate authority
EDI	Electronic Data Interchange
FTP	File Transfer Protocol
GUI	Graphical User Interface
HIDS	Host-based intrusion detection system
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion detection system
IP	Internet Protocol
IPS	Intrusion prevention system
IPsec	IP security
MAC	Media Access Control
MIB	Management information base
MySQL	My Structured Query Language
NAT	Network Address Translation
NIDS	Network intrusion detection system
NIPS	Network Intrusion prevention system
NRPE	Nagios Remote Plugin Executor
NSCA	Nagios Service Check Acceptor
OID	Object identifier
OISF	Open Information Security Foundation

OSI	The Open Systems Interconnection
PEM	Privacy Enhanced Mail
PGP	Pretty Good Privacy
PHP	Hypertext Preprocessor
PKI	Public Key infrastructure
RFC	Request for Comments
S/MIME	Secure/Multipurpose Internet Mail Extensions
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SSL	Secure Sockets Layer
TAP	Test access point
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VoIP	Voice over Internet Protocol
WAP	Wireless Application Protocol

Obsah

1 Úvod

Technologie VoIP se v dnešní době stává velmi oblíbenou díky velké úspoře finančních prostředků, nízkým poplatkům za hovory a odpadajícím poplatkům za vedení pevné telefonní linky. Z velké části je využívána většími společnostmi, ale také úřady, bankami, nemocnicemi a školami. V těchto institucích je nutností, aby byl kladen velký důraz na zabezpečení přenášených informací a dostupnost této služby. Telefonní linka tak již neslouží jen k přenosu samotného hlasu mezi dvěma volajícími, ale k přenosu důvěrných dat jako jsou údaje o bankovních operacích, interních informací společnosti, hesel apod. S rostoucím počtem uživatelů každé služby či technologie spojené s internetem také roste zájem o její ovládnutí, nebo ukořistění informací kybernetickými zločinci.

Můžeme se bránit celou řadou různých technologií, jako jsou různé IDS/IPS a monitorovací systémy, firewally a šifrované přenosy. Samotné IDS/IPS a firewally jsou dostupné nejen v hardwarové podobě, ale i softwarové. Bohužel profesionální hardwarové řešení tohoto problému může být pro malé společnosti a jednotlivce velkou překážkou kvůli jeho vysokým pořizovacím nákladům. Pak se již nabízí druhá zmiňovaná možnost, a to použití softwarových systémů, které jsou ke stažení bez dalších poplatků.

Cílem této práce je snaha prezentovat, a také otestovat možnosti zabezpečení VoIP serveru pomocí současných volně dostupných technologií. V teoretické části představím jednotlivé nástroje a popíšu jejich použití v praxi. Praktická část je věnována jejich otestování. Pro ochranu před síťovým útokem jsou použity nástroje Snort, Suricata a Bro. Monitoring serveru je realizován systémem Zabbix a Nagios. V závěru poukáži jejich výhody a nevýhody a porovnám je s ostatními. Výsledkem by měl být stručný návod na zabezpečení serveru s jádrem linux, obsahující ukázku různých možností ochrany.

2 Teoretický úvod do problematiky zabezpečení serveru na bázi GNU/Linux, správa serveru pomocí protokolu SSH

2.1 Důvody k zabezpečení

Pojem bezpečnost zahrnuje ochranu dat před odcizením, ale i před ztrátou. U první zmíněné ochrany se jedná o útoky v podobě odposlouchávání, sledování hovoru, modifikace dat, viry, spamy, ale také napadení s výsledkem nedostupnosti služby (DDOS útok).

Obávané odposlouchávání hovoru ve skutečnosti není tak jednoduše proveditelné, protože útočník musí mít fyzický přístup k síťovému médiu, po kterém jsou pakety přenášeny. Bezpečnostní opatření serveru jsou vždy kompromisem mezi dostupnými prostředky a dosažením námi žádané úrovně zabezpečení. V první řadě si musíme říct, co a před čím se chceme chránit a jaké jsou naše dostupné prostředky.

Jednou možností jak se bránit, je ochrana proti síťovým útokům (dále jen IDS). Pod zkratkou IDS se skrývá nástroj, který je schopný detekovat útok směřující ze sítě na server. IDS můžou pasivně naslouchat na síti a hledat typicky vypadající znaky útoku. Starají se o monitoring vybraných portů nejen pasivně, ale umí také na hrozbu reagovat například blokováním IP adresy, která se pokouší skenovat porty.

Také ochrana souborů ve firemní síti patří mezi bezpečnostní priority. Pokud se útočníkovi podaří prolomit přístup do systému, je pravděpodobné, že získá i oprávnění uživatele root, pomocí něhož se bude snažit upravit klíčové nástroje, kterými jsou shell a jádro. Proti tomu slouží speciální nástroje pro ověření integrity souborů, které využívají kontrolních součtů.

Za zmínku také stojí ochrana vůči rootkitům. Rootkit je technologie, s níž útočník získá oprávnění root uživatele a může tak mít pohodlný vzdálený přístup a zároveň i ukrytou přítomnost v systému. Pro detekci rootkitů slouží speciální programy. Musíme však počítat s tím, že při napadení systému mohou být napadeny i tyto programy, které by mohli rootkit odhalit.

Nejlepší prevencí je možnost zálohování. Zde je důležité provádět pravidelné zálohy dat, aby byla možná obnova pro případ ztráty. Dále je také třeba provádět zálohu systému, protože obnovení konfigurace systému a služeb je velmi složité a časově náročné.

Monitorování systému a služeb je posledním důležitým pojmem, který bych chtěl zmínit. Na toto téma je zaměřena pátá kapitola v této práci. Je důležité sledovat aktivitu serveru, tzn. pročítat logy. Dají se pročítat ručně nebo se dají zasílat e-mailem již zpracované.

2.2 Správa serveru pomocí protokolu SSH

Secure Shell (dále jen SSH) patří mezi real-time protokoly používáné správci nejen ve VoIP telefonii k údržbě a troubleshootingu.

Můžeme ho popsat jako tunel vytvořený mezi dvěma počítači. Obsahuje sadu tří nástrojů – SLOGIN, SSH a SCP, které nahrazují dřívější RLOGIN, RSH a RCP. Pomocí SSH můžeme získat přístup k shellu druhého počítače, provádět operace se soubory, spouštět příkazy na vzdálených počítačích a směrovat porty. Po nainstalování balíčku sshfs je možné připojit vzdálený souborový systém do místního adresáře. [1]

SSH komunikace pracuje na principu klient – server. Pro bezproblémový chod musíme mít na počítači localhost klient SSH a na počítači server SSHD. Po odeslání příkazu pro připojení na vzdálený server se mezi oběma počítači vytvoří šifrovaný kanál a poté již veškerá komunikace probíhá přes něj. Následuje autentizace pomocí hesla nebo PKI.

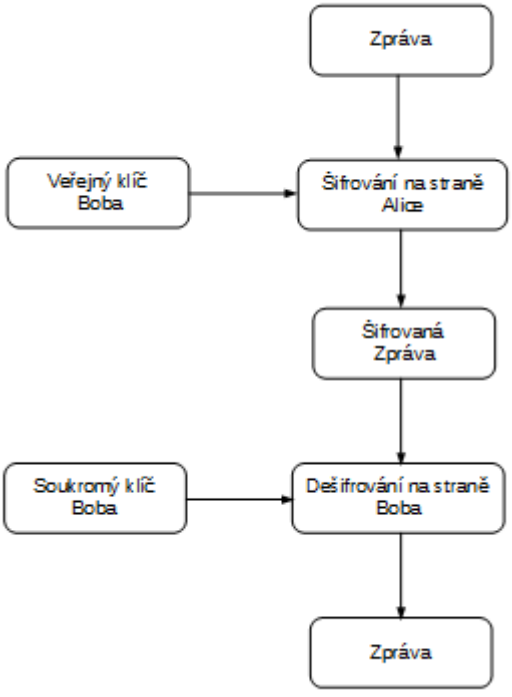
Může dojít k podvržení odpovědi DNS serveru a k připojení k serveru útočníka, kterému tak pošleme heslo. V protokolu je i tato možnost ošetřena, proto nejen server vyžaduje identifikaci po klientovi, ale i klient po serveru. [16]

3 Popis problematiky PKI a její využití na straně serveru

3.1 Public Key Infrastructure

PKI vychází z idey asymetrické kryptografie. V ASK jsou pro šifrování a dešifrování používány dva odlišné klíče narozdíl od symetrické kryptografie, ve které je užíván pouze jeden klíč. Princip PKI je tedy takový, že pokud něco zašifruji jedním klíčem z páru, tak dešifrování je možné pouze druhým klíčem z dvojice. Tato data šifrována mým veřejným klíčem tedy můžu přečíst pouze já. Soukromý klíč je nutné mít na bezpečném místě, zatímco veřejný slouží ke zveřejnění, aby ostatní mohli rozluštit zprávy šifrované mým soukromým klíčem. Po dešifrování mým veřejným klíčem má příjemce jistotu, že právě já jsem tím autorem. Na tomhle principu funguje digitální podpis.

Na obrázku 1 je vysvětlen princip PKI. První z klíčů je veřejný. Ten slouží k šifrování dat určeným k přenosu. Druhým klíčem je klíč soukromý, sloužící k dešifrování dat příchozích. Pokud Alice chce odeslat zprávu Bobovi, použije k zašifrování jeho veřejný klíč. Po doručení Bob zprávu dešifruje pomocí svého soukromého klíče. Pokud chce na zprávu Alici odpovědět, musí znát její veřejný klíč. Výhodou je, že nedochází k přenosu dešifrovacího klíče a nelze jej tedy během přenosu odchytit třetí stranou.



Obr. 1:Princip PKI [2]

3.2 PKI infrastruktura

Pojmem PKI je označována infrastruktura, která se zabývá zabezpečením pomocí šifrování. Základními prvky jsou:

- Certifikační autorita,
- registrační autorita,
- seznam zneplatněných certifikátů,
- OCSP,
- digitální certifikát,
- digitální podpis.

3.2.1 Certifikační autorita

O důvěryhodnost vztahu mezi jedincem a jeho klíčem se stará třetí strana takzvaná certifikační autorita (dále jen CA), která řídí správu kryptografických klíčů a certifikátů. Jejím hlavním úkolem je jejich vydávání, používání a odvolávání. Pokud je tedy námi použitá CA důvěryhodná, můžeme tedy důvěřovat i údajům uvedených v certifikátu, nebo vztahu mezi dvěma klíči.

V takto podepsaném veřejném šifrovacím klíči najdeme údaje o majiteli, za jejichž správnost ručí.

3.2.2 Registrační autorita

Registrační autorita zastupuje autoritu certifikační a ověřuje totožnost žadatele. Přijímá žádosti o nové certifikáty a následně vyřízené certifikáty předává koncovému uživateli. Také má starost zneplatnění certifikátů.

3.2.3 Seznam zneplatněných certifikátů

Seznam zneplatněných certifikátů (dále jen CRL) je databází, do které zveřejňuje CA odvolané certifikáty. K odvolání certifikátu v době jeho platnosti může dojít při ztrátě nebo zcizení párového klíče, popřípadě při porušení podmínek CA a na žádost uživatele, nebo jiné autority. V záznamu je uvedeno číslo certifikátu, datum zneplatnění a důvod. Seznam by měl být veřejně dostupný na internetu nebo přes protokol LDAP.

3.2.4 OSCP

Význam zkratky OCSP se dá vyložit jako online služba pro zjištění aktuálního stavu certifikátu. Přes protokol http můžeme zkontrolovat platnost konkrétního certifikátu, bez nutnosti stahovat celé CRL.

3.2.5 Digitální certifikát

Jako digitální certifikát v ASK slouží veřejný šifrovací klíč vydaný certifikační autoritou. Lze říci, že se jedná o obdobu občanského průkazu, ve kterém jsou obsaženy údaje o jeho majiteli. Nejčastěji jsou používány pro identifikaci uživatele přistupujícího na HTTPS nebo do VPN.

3.2.6 Digitální podpis

Digitální podpis je mechanismus sloužící k potvrzení pravosti dat. Pomocí hashovacího algoritmu se vytvoří otisk dat. Ten je poté zašifrován soukromým klíčem uživatele. Zašifrovaný podpis je pak přiložen ke zprávě. Příjemce stejným hashovacím algoritmem vypočítá otisk příchozích dat. Veřejným klíčem odesílatele dešifruje jeho digitální podpis a porovná hash vypočtený ze zprávy a hash získaný z podpisu. Pokud je shodný, má příjemce jistotu že identita odesílatele je pravá. [3]

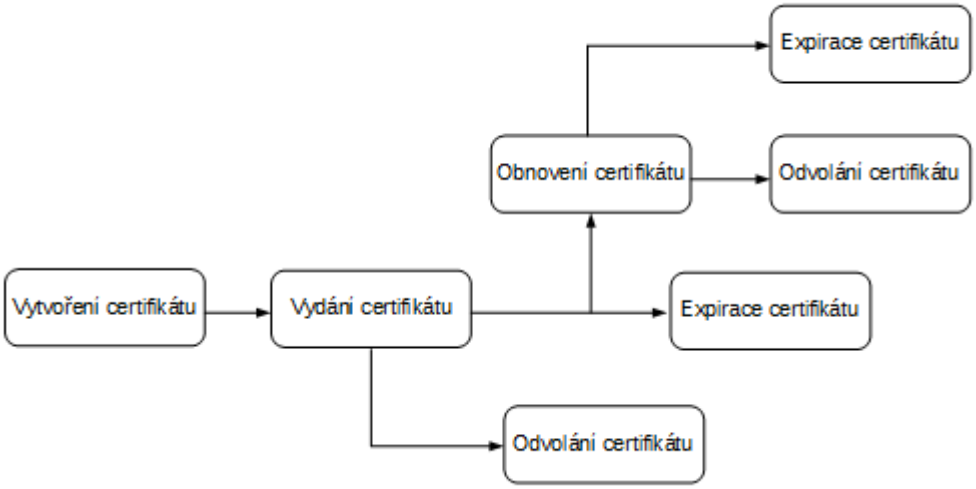
3.3 Služby poskytující PKI

Autentizací rozumíme proces, při němž ověřujeme identitu. V praxi např. kontrola totožnosti uživatele přistupujícího přes portál do systému. Což je spojeno s autorizací, po které jsou přiřazena práva určité osobě, jako je třeba přístup do určitých složek a omezená práce se soubory. Tato funkce je zajištěna digitálním certifikátem.

Druhou vlastností PKI je nepopíratelnost. Všichni uživatelé nesou odpovědnost za své činy. Není možné popřít odeslané data a provedené úkony, ke kterým se vztahuje identita konkrétního uživatele. Tato vlastnost je zajištěna pomocí digitálního podpisu.

Další nedílnou součástí je důvěrnost, která ručí za bezpečný přenos dat přes veřejnou nezabezpečenou síť. Dává nám jistotu, že při přenosu se nedostane k datům neautorizovaná osoba. Pro zvýšení bezpečnosti mohou být použity šifrovací algoritmy.

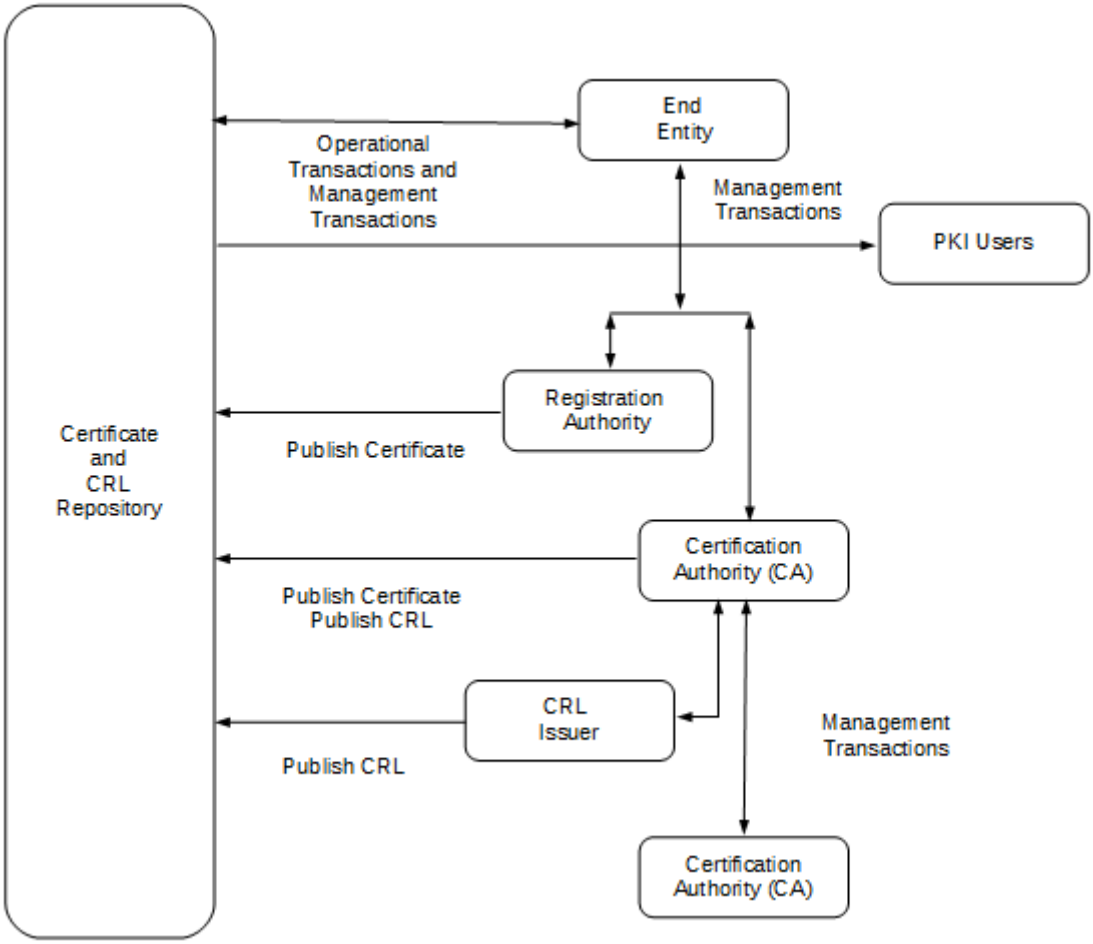
Integrita zajišťuje, že data průchodem veřejnou sítí nebyla změněna ani jinak modifikována. K tomu v PKI slouží hashovaní zpráv. [3]



Obr. 2:Životní cyklus certifikátu

3.4 Architektura modelu PKI :

- Uživatel: koncový uživatel, kterému byl vydán certifikát.
- CA: certifikační autorita vydává certifikáty.
- RA: registrační autorita kontroluje elektronickou žádost a totožnost žadatele.
- CRL: systém pro správu odvolaných certifikátů.
- Repository: Úložný systém shromažďující certifikáty.



Obr. 3:Architektura PKI[2]

3.5 Standardizace

Strukturu certifikátu PKI specifikuje několik standardů, které určují jaké techniky, formát a syntaxe se při šifrování mají používat.

Nejpoužívanějším pro vydávání certifikátů v dnešní době je standard X.509 ve verzi 3, popsaný v doporučení RFC-5280. Každý takový certifikát musí obsahovat alespoň tyto parametry:

- Veřejný klíč,
- identifikaci vlastníka veřejného klíče,
- identifikaci CA,
- podpis CA,
- číslo certifikátu,
- verzi standardu X.509.

U protokolů S/MIME ,SSL a IPsec slouží pro autentizaci uživatelů a pro výměnu šifrovacích klíčů. Za zmínku stojí i jiné typy certifikátů jako jsou EDI, WAP, PGP, PEM.

3.6 Autentizace pomocí klíče při přihlášení k serveru

PKI neslouží jen jako jedno z řešení pro šifrování, podpis elektronické pošty nebo přihlašování k doméně. Oblast PKI je mnohem obsáhlejší. Další možností může být zajištění nepopiratelnosti prováděných transakcí, zajištění důvěrnosti, integrity a umožnění autentizace osob.

Nejčastějším prostředkem pro autentizaci je heslo. Toto řešení má však řadu nevýhod. Je možné jej odposlechnout, uživatel si ho musí pamatovat a musí být dostatečně složité, aby bylo obtížné jej uhodnout. Všechny tyto nevýhody lze vyřešit použitím metody SSH pomocí PKI.

3.6.1 SSH pomocí PKI

Vygenerujeme pár klíčů, veřejný nahrajeme na server, kam se budeme chtít přihlašovat a soukromý si ponecháme. Následné přihlášení funguje tím principem, že server zašle náhodná data. Uživatel data podepíše soukromým klíčem a pošle zpět serveru. Server ověří pravost veřejným klíčem a povolí přístup. Nedochází tak k přenosu žádného hesla, ani soukromého klíče.

Vygenerování klíče:

```
ssh-keygen -t rsa
```

Oba vytvořené klíče jsou uloženy do adresáře /.ssh pod názvy id_rsa a id_rsa.pub. Kde id_rsa.pub je veřejný.

```
Your identification has been saved in
/home/ondroid/.ssh/id_rsa.

Your public key has been saved in
/home/ondroid/.ssh/id_rsa.pub.
```

Veřejný klíč teď nahrajeme na server do souboru .ssh/authorized_keys.

```
cat id_rsa.pub >> .ssh/authorized_keys
```

Pro zvýšení bezpečnosti, zakážeme přihlašování pomocí hesla v konfiguračním souboru. A také se ujistíme, že je povoleno přihlášení přes SSH. V souboru, který nalezneme zde:

/etc/ssh/ssh/sshd_config.

```
RSAAuthentication yes
PubkeyAuthentication yes
ChallengeResponseAuthentication no
```

```
PasswordAuthentication no
UsePAM no
```

Tímto jsme docílili bezpečnějšího přihlašování, ale také zamezili tomu, aby útočník mohl zkoušet heslo uhodnout. Změny se projeví až po restartování SSH serveru, což provedeme příkazem:

```
/etc/init.d/ssh reload
```

4 Nasazení firewallu a použití IDS/IPS systémů

4.1 Firewall

Firewall je bezpečnostní systém, rozdělující jednu síť od druhé. Z pravidla umístěný mezi privátní a veřejnou sítí, obsahující pravidla, podle kterých je rozhodnuto, který síťový tok bude propuštěn, odmítnut nebo blokován. Tuto funkci nejčastěji vykonává router, počítač se specialním systémem, nebo jiné síťové zařízení schopné filtrování paketů.[4]

4.2 Kategorie firewallu

4.2.1 Paketové filtry

Nejstarší a nejjednodušší variantou firewallu je paketový filtr. Uvádí z jaké adresy a portu na jakou adresu a port může být paket doručen. Paket je kontrolován na třetí a čtvrté vrstvě OSI. Paketové filtry se vyznačují vysokou rychlostí zpracování, nevýhodou však je nízká úroveň kontroly.

4.2.2 Aplikační brány

Aplikační brány neboli také proxy firewally zcela oddělují jednotlivé sítě. Komunikace probíhá spojením dvou bodů přes aplikační bránu. Klient, inicializující komunikaci, se připojí na proxy bránu, která spojení zpracuje a otevře nové připojení k požadovanému serveru. Odpověď putuje zpět stejnou cestou pomocí původního spojení. Server tak nevidí zdrojovou adresu klienta – jako zdroj figuruje adresa aplikační brány. Proxy firewall tak automaticky plní funkci překladače adres. Výhodou je vysoká úroveň zabezpečení. Nevýhodou však vysoká hardwarová náročnost. Používají se tam, kde nestačí jen pouhé filtrování paketů, ale je potřeba i hloubkového dohledu nad přenášenými daty. Kontrola se tak provádí na aplikační vrstvě modelu OSI.

4.2.3 Stavové paketové filtry

Jedná se o rozšíření standardních paketových filtrů, o možnost rozlišení již používaného spojení od nově příchozích. Porovnává tak tabulkové definice, ale také udržuje přehled již probíhající komunikace. Díky tomu je možné reagovat na ustanovená spojení a automaticky povoluje odpověď.

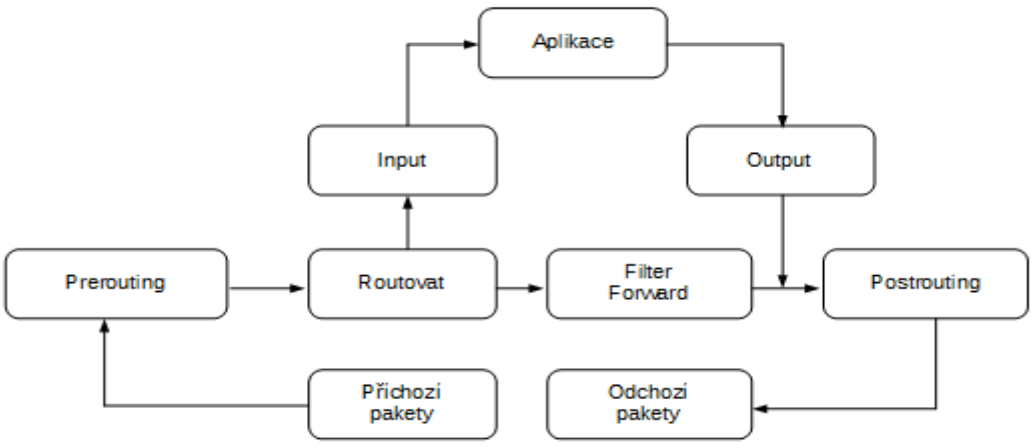
Tím pádem už nerozhoduje jen podle zdrojové a cílové adresy, kterou obsahuje paket, ale i na základě parametrů probíhajícího spojení. Používají se čtyři základní stavy a to NEW, ESTABLISHED, RELATED a INVALID. Kde NEW označuje nově začínající komunikaci, ESTABLISHED komunikaci ustanovenou, kde je očekávána odpověď. RELATED pak značí stav spojení, které by mělo zůstat otevřeno z důvodu očekávané odpovědi. Poslední stav INVALID slouží

pro paket s neodpovídajícím očekávaným chováním. Taktéž jako paketový filtr pracuje na třetí a čtvrté vrstvě modelu OSI.

4.3 Netfilter

Jedná se o Linuxový firewall, který je již implementován v jádře. Konfigurace je možná přes program Iptables. Pracuje na principu stavového filtru, dokáže tak analyzovat spojení a ne jen samotný paket. Využíván je také protokolem NAT, který taktéž nutně potřebuje vědět o stavu probíhajících spojení. K vysvětlení principu Netfilteru použitého pro zabezpečení VoIP serveru postačí jeho zjednodušené blokové schéma. Netfilter je složený celkově ze čtyř částí, a tím jsou filter, nat, mangle a raw. Ve filteru dochází k samotnému filtrování paketů podle třech třech pravidel ACCEPT, DROP a REJECT. Každá ze čtyř výše jmenovaných tabulek ještě obsahuje řetězce typu INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING, podle kterých je rozhodnuto kam paket bude směrovat. INPUT a OUTPUT značí pakety příchozí a odchozí z lokální stanice. FORWARD pravidlo předává paket mezi jednotlivými síťovými rozhraními. PREROUTING obsahuje sadu pravidel pro změnu cíle, než paket projde firewallem a POSTROUTING naopak pravidla pro změnu zdroje již po průchodu firewallem.

Princip Netfilteru je tedy následující, když na vstup přijde paket, je řetězcem PREROUTING rozhodnuto, zda patří místní stanici. Pokud ne, je automaticky předán na rozhraní, aby pokračoval do jiné sítě. Prochází FORWARD řetězcem, kde probíhá filtrování mezi jednotlivými sítěmi. Jestliže paket patří místní stanici, pokračuje k řetězci INPUT. V momentě, když je zpracována odpověď, je odevzdána řetězci OUTPUT. Po zpracování řetězcem OUTPUT nebo FORWARD je předán paket pro vykonání řetězci POSTROUTING. [5]



Obr. 4:Architektura netfilteru [4]

4.4 Nasazení firewallu

Zde si ukážeme, jak lze v programu Iptables vytvořit nové pravidlo a vysvětlíme jednotlivé části jeho struktury.

Syntaxe pro přidání nového pravidla:

iptables [tabulka] [akce] [řetězec] [pravidla] [cíl]

Tabulka může nabývat hodnot:

- Filter – filtrování paketů
- Nat – překlad adres
- Mangle – zpracování hlaviček paketů

Pokud není pole tabulka vyplněno, nabývá automaticky hodnotu filter. Následuje akce, která nabývá hodnot:

- -A – přidání nového pravidla na konec řetězce
- -P – zadání hlavního pravidla
- -L – vypíše pravidla
- -F – vymaže pravidla
- -I – přidání nového pravidla na začátek řetězce
- -D – smaže pravidlo
- -N – vytvoří nový řetězec
- -X – smaže námi vytvořený řetězec
- -E – přejmenuje řetězec

Následuje pole řetězec, pro tabulku filter může nabývat těchto hodnot:

- INPUT – vstupní řetězec, pro vstupující pakety do počítače
- OUTPUT – výstupní řetězec, pro odchozí pakety z počítače
- FORWARD – určuje, co se provede s pakety, které nejsou určeny pro daný PC

Řetězec pro tabulku nat nabývá těchto hodnot:

- PREROUTING – používá se pro změnu cílové adresy
- POSTROUTING – používá se pro změnu zdrojové adresy

Zde jsou nejpoužívanější hodnoty pro pole pravidla:

- -s – zdrojová IP paketu
- -d – cílová IP paketu
- -i – vstupní zařízení, kterým bude paket přijat např. eth0
- -o – výstupní zařízení, kterým bude paket odeslán
- --sport – zdrojový port paketu
- --dport – port na který paket putuje

Pole cíl určuje, co bude s paketem provedeno. K určení cíle je použit parametr -j za kterým následuje jedna z těchto akcí:

- ACCEPT – paket je akceptován, a může projít filtrem
- REJECT – paket je zahozen a zdrojový PC je o této akci informován
- DROP – paket je zahozen a zdrojový PC není o této akci informován

Zde je již několik navržených pravidel pro hlavní protokoly používané ve VoIP telefonii:

```
# SIP
 iptables -A INPUT -p udp -m udp --dport 5060 -j ACCEPT

# IAX2
 iptables -A INPUT -p udp -m udp --dport 4569 -j ACCEPT

# IAX
 iptables -A INPUT -p udp -m udp --dport 5036 -j ACCEPT

# RTP
 iptables -A INPUT -p udp -m udp --dport 10000:20000 -j ACCEPT

# MGCP
 iptables -A INPUT -p udp -m udp --dport 2727 -j ACCEPT
```

[6]

4.5 Úvod do IDS a IPS

Systém detekce útoků (IDS) a Systém Prevence proti útokům (IPS) jsou nástroje pro zajištění bezpečnosti systému ale i sítě. Jedná se o softwarově i hardwarově řešené systémy provádějící inspekci paketů až po aplikační vrstvu modelu OSI.

4.6 IDS

IDS je jen jednou z více částí ochranného systému, není tak samostatným bezpečnostním opatřením. Slouží pouze k detekování a upozornění na podezřelé aktivity v síti, které můžou, ale i nemusí být hrozbou. Tyto upozornění jsou uložena do logu nebo jsou přijata dalšími podpůrnými systémy, které již provedou příslušná opatření. Mezi spolupracující systémy patří IPS a firewally, které se těmto útokům již snaží bránit. IDS lze do sítě připojit několika způsoby. Můžeme jej připojit na rozhraní síťového zařízení, jako je router, switch, nebo firewall. Každý paket je pak zkopírován a prověřen. Dále pak pomocí zařízení TAP, které pracuje jako sonda, skrz kterou protéká provoz určený pro analýzu. Poslední jmenovanou možností je připojení ethernetového hubu, který tak procházející data rozesílá na ostatní porty, na jehož konci je připojeno IDS.

IDS detekují potencionální útok na základě činnosti jednoho ze dvou principů:

Porovnáním vzorů

Detekce narušení prováděná porovnáváním s pravidly (signaturami) uloženými v databázi, které mají podobu některého známého útoku. Databáze pravidla již obsahuje, ale je možné i dopisování vlastních vzorů. V šesté kapitole jsou přiloženy ukázky pravidel pro každý ze tří IDS systémů.

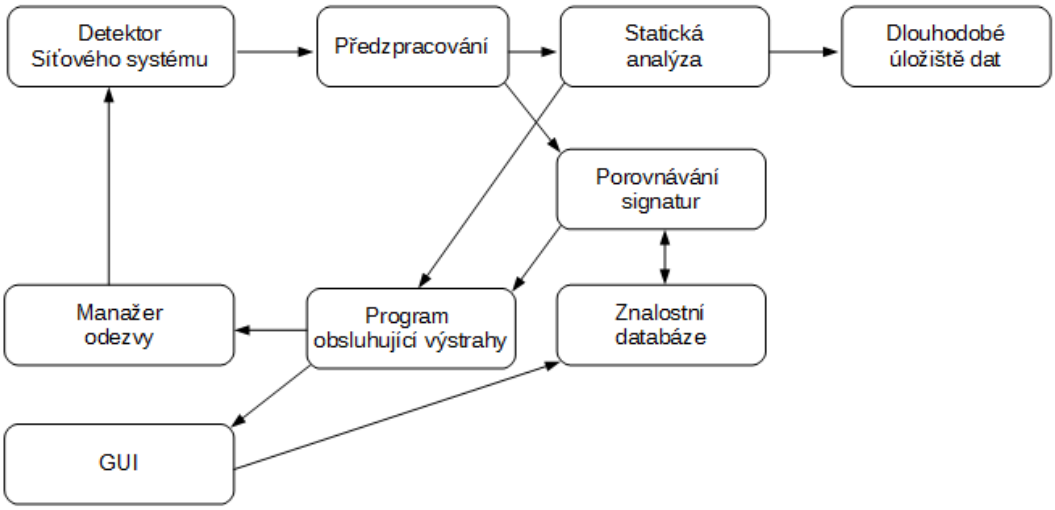
Detekováním statické anomálie

Systém analyzuje provoz v síti a pátrá po výjimečném stavu, nebo jakékoliv odchylce od normálu. Sledováním sítě po delší časové období se stanoví normální stav. Upozorněno je pak na cokoliv co vykazuje nestandardní chování. Například pokud se třeba rapidně zvýší počet dotazů na server, je událost považována za útok. Výhodou je detekce dosud neznámých útoků, takový útok je odhalen, pokud v komunikaci dojde k odklonu od určitého standardu.

Vytváření modelu normálního chování lze rozdělit na staticky a dynamický model. Statický model chování je vytvořen pouze jedenkrát a dále již je neměnný. Toto sestavení je využíváno systémy, které v čase vykazují stejné chování. Do modelu dynamického jsou však průběžně přidávány nové záznamy o chování systému, samotný systém je tak flexibilnější. Nevýhodou obou modelů je, pokud je systém pod útokem ve fázi vytváření normálního stavu a dojde-li ke stejnému útoku, server nebude vykazovat známky neznámého chování a nemusí tak být útok odhalen.

Důležitý faktor je čas, potřebný k vyhodnocení, zdali se jedná o skutečné napadení. Vyhodnocování v reálném čase dává možnost téměř ihned reagovat na odhalený útok, ale při větším zatížení sítě je nutné počítat s vyššími nároky na výkon stanice. Zatímco u intervalově orientovaného přístupu jsou informace ukládány periodicky a posléze vyhodnocovány. Výhodou jsou nízké nároky na výpočetní výkon. Využíváno pouze v prostředí s nízkým rizikem napadení a nízkými

potencionálními ztrátami. Chybí však bezprostřední reakce na útok a nutností je dostatečně velká paměť pro ukládání záznamů.[7]



Obr. 5:Schéma IDS [4]

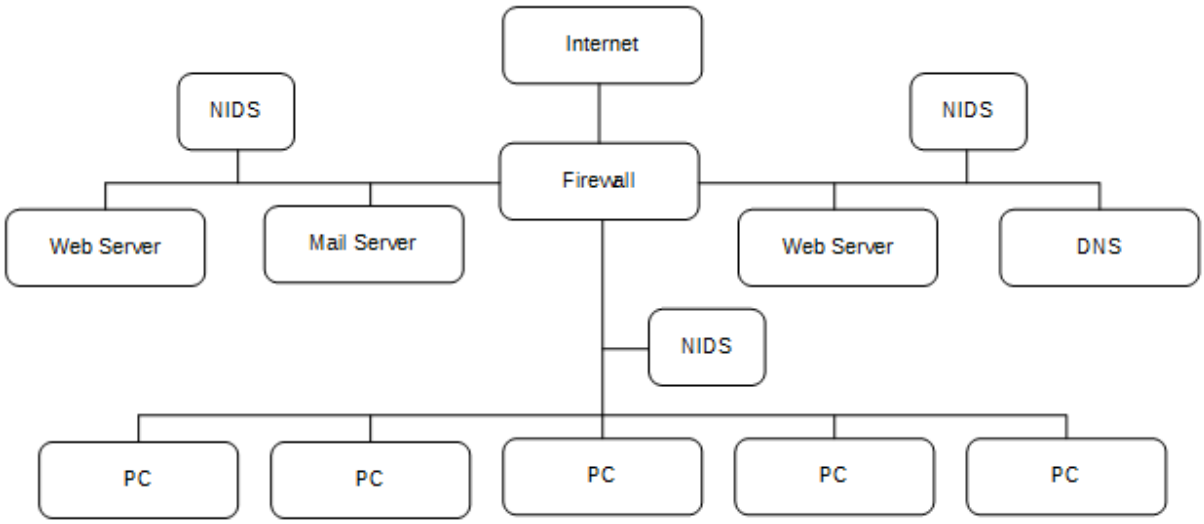
IDS můžeme rozdělit do tří základních skupin podle jejich umístění v síti:

- NIDS(Network Based Intrusion Detection Systems) - síťově orientované systémy detekce narušení
- HIDS(Host Based Intrusion Detection Systems) – hostitelsky orientované systémy detekce narušení
- DIDS (Distributed Intrusion Detection System) – distribuované systémy detekce narušení

4.6.1 NIDS

Síťová IDS jsou umístěny na strategickém segmentu námi monitorované sítě a kontrolují každý procházející paket. Stadartně síťová karta pracuje v nepromiskuitním režimu, ve kterém jsou analyzovány pouze pakety směrované pro MAC (Media Access Control) této karty. Je nutné, aby karta byla nastavena do režimu promiskuitního módu, ve kterém je umožněno zachytávání celého síťového provozu. Při větším vytížení sítě roste objem dat ke zpracování. Kontrolou velkého množství pravidel rostou i hardwarové nároky a může docházet k přetížení. Tím se zvyšuje šance, že útok nebude odhalen. Může také dojít k výskytu falešných útoků, které je poté obtížné rozlišit od reálných hrozeb. Je tedy nutné mít správně nastavený systém a omezit falešné útoky na minimum, jinak může dojít k ignorování alertů a tím rapidně klesá bezpečnost. Mezi nejznámější systémy patří Suricata, Bro, Snort, Firestorm a Tamandua. [8]

Na obrázku 6 je zobrazena síť, obsahující tři NIDS. Jsou rozmístěny tak, aby bylo možné monitorovat veškerý síťový provoz v dané topologii.

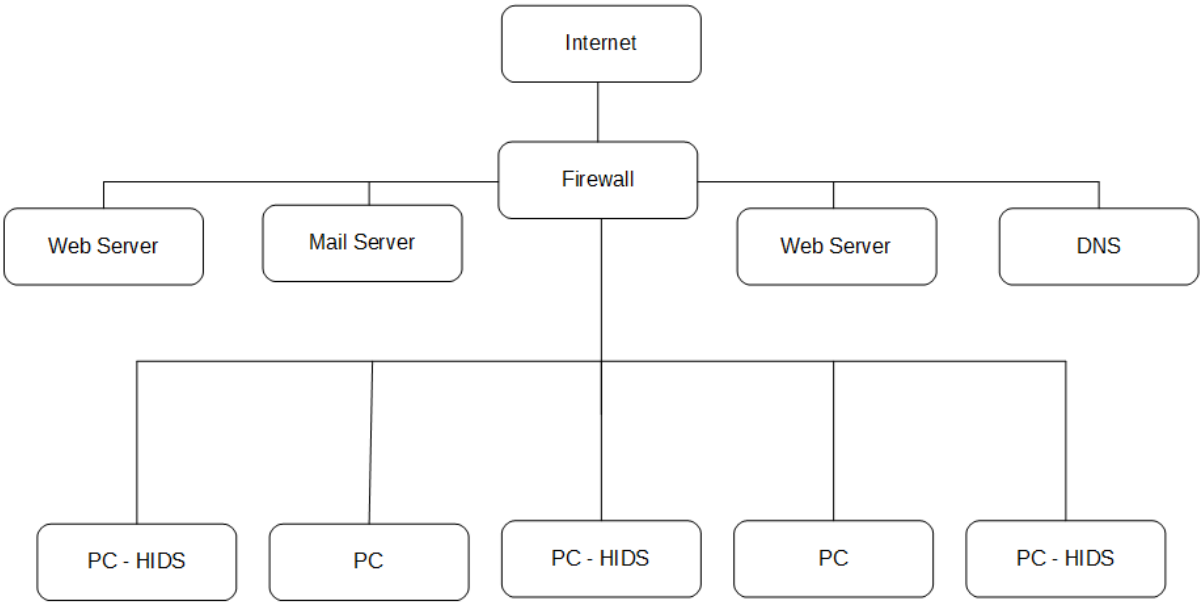


Obr. 6:Schéma NIDS [8]

4.6.2 HIDS

Systém HIDS je zaměřen pouze na bezpečnost hostitelského systému. Je schopný kontrolovat aktivitu uživatelů v síti i probíhající změny v souborovém systému. Vyhodnocuje tyto procesy u hostitele tak, že prochází obsah logů a porovnává, jestli se některý neshoduje s obsahem z databáze signatur útoku. Nutná je instalace na každé zařízení zvlášť. Máme možnost řídit každý zvlášť nebo vzdáleně přes server. Síťová karta stanice pracuje v nepromiskuitním módu, který je náročný výpočetní výkon, z čehož plyne omezení využití na slabších stanicích. Snížením počtu pravidel klesá objem vykonaných instrukcí, tímto můžeme také snížit potřebný výpočetní výkon stanice. Pak lze systém provozovat i na slabším počítači, ale na úkor bezpečnosti. HIDS pracují na aplikační vrstvě modelu OSI, dokáže na rozdíl od systému NIDS analyzovat šifrovaný přenos dat a detekovat útoky vedené přes šifrovaný kanál. Nevýhodou je možnost vyřazení z provozu pomocí Dos útoků. Použít můžeme např. Tripwire, Snortinline, LIDS. [8]

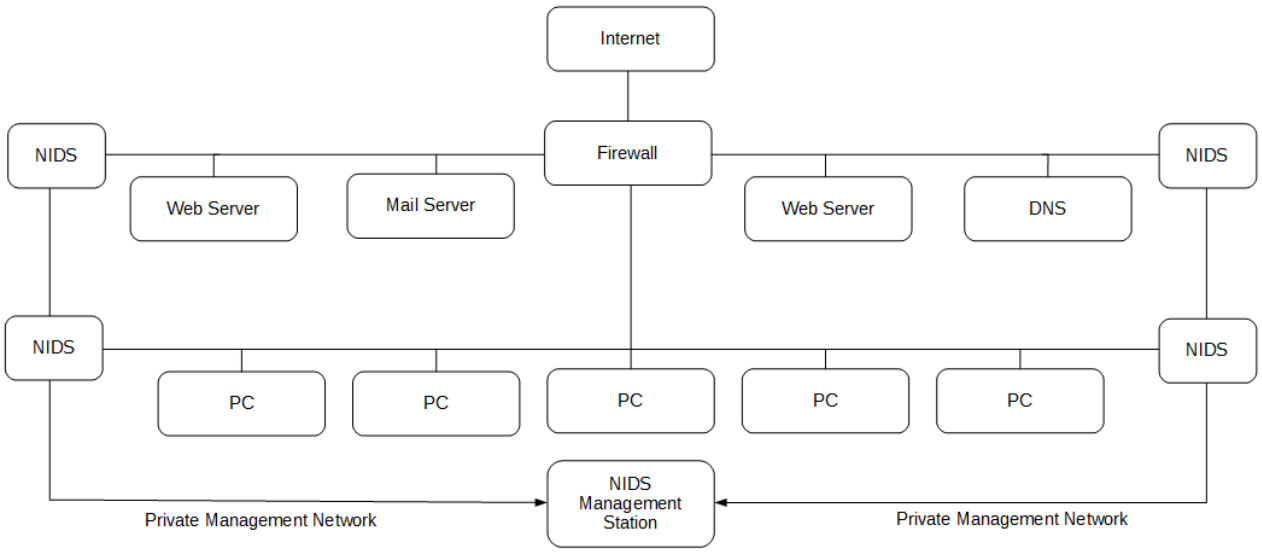
Na obrázku 3 je zobrazena topologie, která obsahuje systém HIDS na hostitelských stanicích.



Obr. 7:Schéma HIDS [8]

4.6.3 DIDS

Systém je kombinací předchozích dvou zmíněných systémů NIDS a HIDS, dosahuje tak nejlepších výsledků díky využití výhod obou a částečné eliminaci jejich nevýhod. Architektura by se dala popsat jako sonda-manažer. Kdy systémové i síťové sondy jsou rozmístěny kdekoliv v síti a jsou monitorovány centrálně. Tím pádem je monitorován provoz sítě i jednotlivé systémy. Není tedy nutné mít všechny sondy jen typu NIDS, nebo HIDS. Možné je mít kombinaci obou typů, dokonce i různých výrobců. Teoreticky takto navržený systém by měl zajistit zcela kompletní detekci útoků. Síťové karty tak můžou pracovat v obou režimech, jak v promiskuitním, tak nepromiskuitním.[8]



Obr. 8:Schéma DIDS [8]

4.7 IPS

IPS (Intrusion Prevention Systems) systémy pracují na podobném principu jako IDS. Umožňují útok detekovat, ale také podniknout kroky k jeho zabránění. IPS je do sítě nasazen jako aktivní člen analyzující probíhající události.

4.7.1 HIPS (Host IPS) – uzlové

HIPS lze definovat jako nástroj pro sledování jednotlivého hosta a údálostí jím vykonaných. Zpracovává všechny požadavky na systém, nesmí však výrazně zatěžovat výpočetní výkon stanice, ani jinak ovlivňovat její provoz.

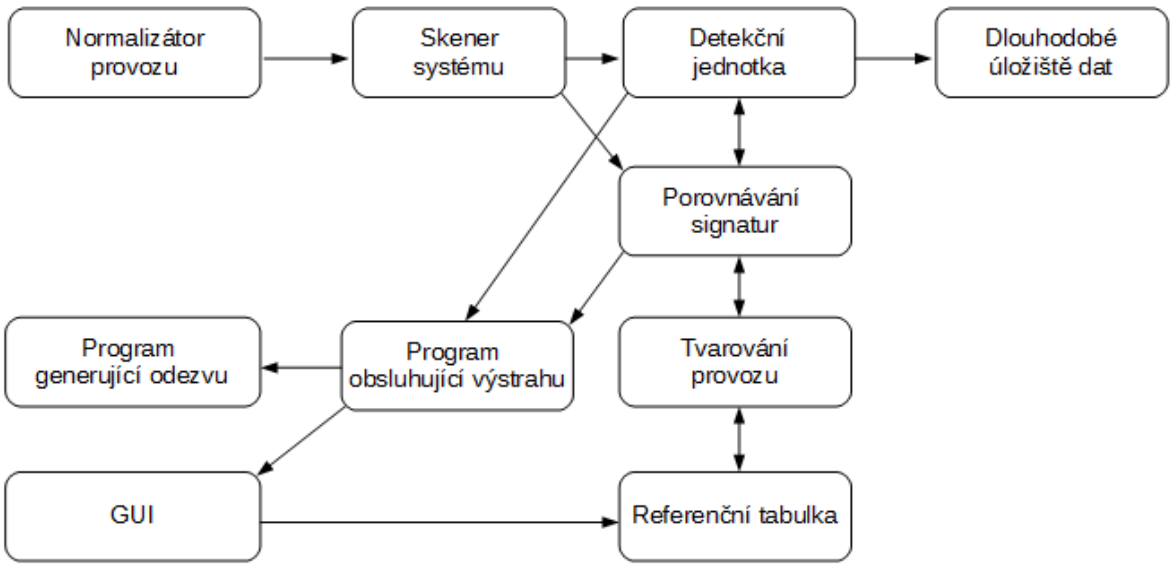
4.7.2 NIPS (Network IPS) – síťové

Je kombinací IDS, IPS a firewallu. NIPS má dvě a více síťových rozhraní, která jsou označeny jako interní a externí. Pakety přicházející na rozhraní jsou analyzovány, aby se ověřilo, zda se jedná o hrozbu či ne.

4.7.3 Schéma IPS systému

Typické IPS je složeno z následujících čtyř částí. Normalizátor provozu, monitor služeb, detekční jednotka a tvarovač.

Normalizátor provozu detekuje nejasnosti či podezřelé pakety protékající čidlem. Zároveň se stará o správné řazení a opětovné skládání paketů. Následně signál přichází do detekční jednotky s monitorem služeb. Úkolem Monitoru služeb je generovat referenční tabulku. Podle níž je informace poté vyhodnocena. Na základě tohoto výsledku pak tvarovač provozu dále řídí tok informací. Dále za pomoci referenční tabulky hledá detekční jednotka signálové vzory, z nichž jsou následně vyhodnocena příslušná opatření. [9]



Obr. 9: Schéma IPS systému[4]

4.8 Snort

Jde o open-source bezpečnostní nástroj pro detekci a prevenci útoků pracující na principu IDS a IPS. Jeho úkolem je hledat signatury známých útoků v síti. Jádro snortu je nápsáno programovacím jazykem C a k zachytávání paketů spolupracuje s knihovnou libpcap. Kontroluje veškerou průchozí komunikaci běžící na protokolech TCP, IP, UDP a ICMP. Obsah paketů je porovnáván s pravidly. Provozovat ho lze na operačních systémech Linux, FreeBSD , OpenBSD, NetBSD, Windows, MacOS X, Solaris a dalších. Hardwarové požadavky na stanici jsou závislé na velikosti monitorované sítě a na objemu provozu. Data z většího množství sond jsou ukládána na jednu stanici, pro jednu sondu je nutné rezervovat okolo 10 GB místa na disku. Pokud nejsou některé pakety zpracovány, příčinou může být nedostatek výkonu stanice. [8]

4.8.1 Režimy Snortu

Sniffer mode

V režim slídiče Snort zachytává všechny průchozí pakety a zobrazuje je uživateli na obrazovku. Spustíme jej pomocí příkazu: snort -v.

Packetlogger mode

Jedná se o rozšířenou verzi sniffer módu. Zachycená data jsou uložena na pevný disk do log souboru. Spustíme jej pomocí příkazu: snort –ved –l ./log.

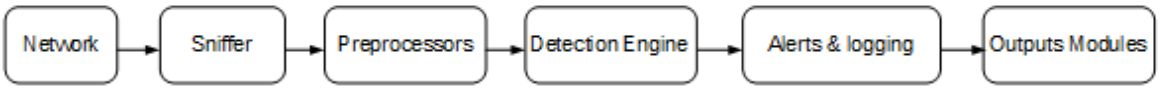
Network intrusion detection systém mode

Režim detekce narušení sloužící k monitorování nebezpečných aktivit v síti. Rozdíl mezi dvěma předchozími módy je ten, že nezaznamenává všechnu aktivitu, ale jen tu, kterou podle pravidel označí jako nebezpečnou. [8]

Spustíme jej pomocí příkazu: snort -c /usr/local/snort/etc/snort.conf -l /var/log/snort

Při zapnutí zadáváme cestu ke konfiguračnímu souboru, ve kterém jsou definována pravidla.

4.8.2 Achitektura IDS Snort



Obr. 10: Architektura Snortu [3]

Pomocí jednotky Sniffer aplikace naslouchá síťovému provozu. Pracuje s protokoly TCP, IP, UDP, ICMP a interpretuje zachycené pakety do pro lidi srozumitelné formy nebo posílá na výstup k dalšímu zpracování.

Preprocesor pomáhá k identifikování potencionálních útoků a připravuje pakety pro detekční jednotku. Dalším úkolem je defragmentace paketů, protože při přenosu větších dat jsou pakety rozděleny na menší díly a pro kontrolu je nutné je zkompletovat.

Detection engine porovnává přijaté data z preprocesoru s databází pravidel. Při zjištění podobnosti se signaturou z databáze jsou data předána jednotce Alert generation (systém logovaní a výstrah).

Jednotka Alerts & logging vyvolá poplach při obdržení zprávy z detekční jednotky. Data můžou být odeslána e-mailem, uložena do SQL databáze, nebo zobrazena na webu. Standardně jsou uložena do log souboru, který se obvykle nachází v /var/log/snort.

4.8.3 Signatury Snortu

Jádrem celého systému jsou pravidla. Jejich jazyk je velmi intuitivní a lze si ho snadno osvojit. Pravidlo je složeno z hlavičky a volby. Z hlavičky můžeme určit, jaká akce bude vykonána, protokol, zdrojovou a cílovou adresy s portami. Volba obsahuje zprávu s informacemi daného paketu.

Formát pravidla:

```
akceprotokolzdroj_ipzdroj_portsměrcíl_ipcíl_port (volby)
```

Ukázka použitého pravidla:

```
alert ip any any -> any 5060 (msg:"VOIP SIP INVITE message
flooding"; content:"INVITE"; depth:6; threshold: type both,
track by_src, count 1000, seconds 60; classtype:attempted-dos;
sid:100000158; rev:2;)
```

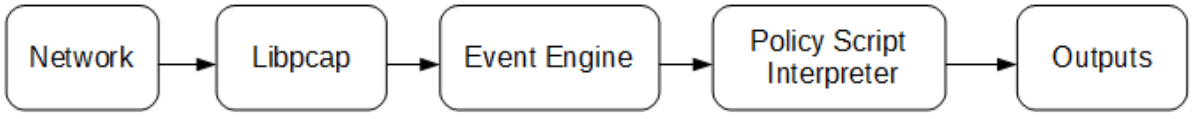
- msg //hlášení pro výstrahu
- content //pátrá po vzoru v paketu
- depth //hloubka hledání v datové části paketu
- threshold //bližší specifikace pravidla
- classtype //klasifikace události
- sid // jednoznačna identifikace pravidla
- rev //číslo revize pravidla

4.9 BRO

BRO je open-source NIDS pracující na systému Unix. Vyvinut Vernonem Paxsonem v Lawrence Berkeley National Laboratory (LBNL) v roce 1996. V této laboratoři běží nepřetržitě již deset let jako hlavní bezpečnostní prvek. Slouží k pasivní více vrstvé analýze a detekci podezřelých aktivit v síti. Je schopný detekce specifických útoků na základě předem definovaných signatur, tak i výskytu neobvyklých činností. Primárně však byl vyvinut pro analýzu a výzkum ve vysokorychlostních sítích.

Podle sémantiky v aplikační vrstvě jsou hledány podobné signatury známých hrozeb. BRO obsahuje již existující skripty, ale k dispozici je i možnost psát skripty vlastní ve speciálně vytvořeném jazyce pro Bro. Velmi nápomocný může být již nástroj snort2bro, který dokáže importovat signatury ze snortu a tím tak rozšířit naši databázi pravidel. Vznikl jako platforma sloužící pro analýzu datového provozu a výzkum detekce průniků. Vyvíjen byl především pro unixové odborníky zabývající se bezpečností. Jeho velkou předností je efektivita a vysoký výkon. Nevýhodou je složitý skriptovací systém a chybějící kvalitní dokumentace. Ovládání je pouze konzolové a využívá jen zápis do textového logu. [9]

4.9.1 Architektura IDS BRO



Obr. 11: Architektura Bro

Architektura systému Bro se skládá z knihovny Libpcap, Event Engine, Policy Script Interpreter. Bro zachytává pakety za pomoci knihovny libpcap, která obsahuje sadu filtrovacích pravidel. Je také využívána programem tcpdump a umožňuje offline analýzu. Používání této knihovny přináší možnost základního filtrování paketů, lze tak analyzovat jen vybranou část provozu. Takto odfiltrovaná část od nedůležitých elementů, pak putuje do jednotky zvané Event Engine, kde je kontrolována integrita a proveden kontrolní součet. Pokud proběhne s negativním výsledkem, Bro vygeneruje událost a paket zahodí. Policy Script Interpreter kontroluje, zda předchozí jednotka vygenerovala nějakou událost. Pokud ano spustí k ní odpovídající skripty. [9]

Ukázka signatury v jazyko Bro:

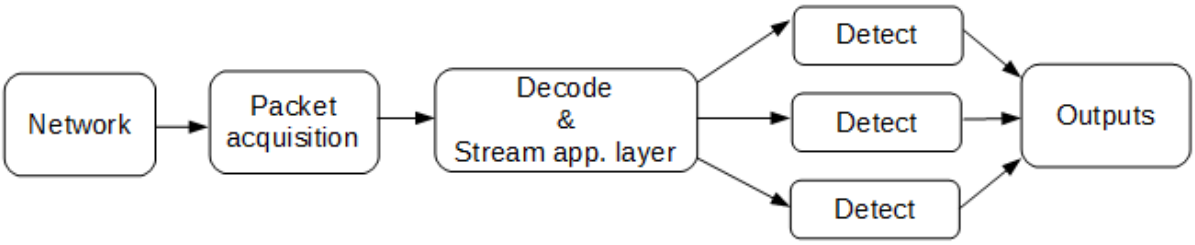
```
signature my-first-sig {
ip-proto == tcp
dst-port == 80
payload /.*root/
event "Found root!"
}
```

4.10 Suricata IDS/IPS

Suricata je výkonný open-source IDS/IPS vyvinut a podporován nadací OISF. Velkou výhodou oproti Snortu je podpora multi-threadingu, automatická detekce protokolů, lepší viditelnost provozu na aplikační vrstvě a rychlejší normalizace a parsování http datového toku. Tak jako Bro i Suricata dokáže převzít již existující sadu pravidel ze Snortu. Pracuje pod systémy FreeBSD, Linux, UNIX, Mac OS X a Windows.

Vzhledem k tomu, že nejvíce výpočetní práce je nutné pro samotnou detekci, rozhodli se vývojáři Suricaty implementovat vláknové rozdělení zátěže pro navýšení výkonu detekce. Samotná analýza je tak rovnoměrně rozdělena mezi jednotlivá jádra a jde o razantní navýšení efektivity. Díky velkému vypočetnímu výkonu dnešních grafických procesorů, lze také využít technologie CUDA a OpenGLk řešení jednoduchých instrukcí v Suricatě. Jádro Suricaty pracuje společně s HTP knihovnou, která provádí parsování a analýzu posloupnosti prvků. Snaží se tak určit gramatickou strukturu vůči dané formální gramatice. Hlavní funkcí tohoto modulu je analyzovat data protokolu http. [10]

Na obrázku 12 je schéma architektury Suricaty, kde je detekce prováděna ve třech samostatných vláknech.



Obr. 12: Architektura Suricata

5 Monitoring serveru a popis specifických požadavků VoIP serveru

5.1 Zabbix

Nástroj Zabbix slouží ke sledování širokého spektra síťových služeb, serverů, počítačových stanic a mnoho dalších prvků. Vytvořen je v jazyce C a rozhraní v PHP. Velkou výhodou je možnost sledování dostupnosti služeb HTTP a SMTP bez potřeby instalace softwaru na sledované stanici. Odpadá tak nutnost instalace agenta, díky možnosti využití monitoringu pomocí protokolu SNMP. Zabbix agent je možné instalovat na stanici na stanici se systémem UNIX i Windows, poté můžeme sledovat informace jako je využití sítě, zatížení procesoru, nebo volné místo na disku. Disponuje funkcí ověření dostupnosti webových stránek, ale také může kontrolovat jejich obsah. Velkou nevýhodou Zabbixu je absence možnosti instalování rozšiřujících pluginů. Pro monitoring méně rozsáhlých sítí je tento systém dostačující.

5.2 Nagios

Nagios je open source nástroj určený k monitoringu systému. Jeho úkolem je sledovat stav určitého zařízení a služby na něj běžící. Pokud se na síti vyskytne jakákoliv událost, úkolem Nagiosu je informovat o ní správce co nejrychleji, ještě dříve než ji zaznamená sám uživatel. Objekty určené k monitoringu můžeme rozdělit dvou skupin, hosts a services. Hosti jsou fyzická zařízení jako routery, servery, počítače, nebo tiskárny. Služby jsou funkce běžící na hostovi například v podobě webového serveru, které je naším úkolem monitorovat. [11]

5.3 Centreon

Centreon je systémový, síťový a aplikační sledovací nástroj postavený na jádru Nagiosu. Byl vyvíjen jako nadstavba Nagiosu, ale přidáváním dalších technických prvků a doplňků se z něj stal plnohodnotný monitorovací systém. Nastavení a získaná data jsou ukládána do MySQL databáze a konfiguračních souborů Nagiosu. Oproti Nagiosu přibyla velmi užitečná funkce zobrazení sítě pomocí 3D mapy.

5.4 SNMP Protokol

SNMP protokol navržený v 80. letech, spadá do sady internetových protokolů sloužících pro monitoring sítí. Zároveň umožňuje sběr dat pro potřeby správy sítě. Podporován velkým množstvím zařízení, jako jsou aktivní síťové prvky, přístupové body, počítače, servery, tiskárny atd. Prostřednictvím SNMP je dnes realizována většina monitorovacích nástrojů. SNMP je založen na

architektuře klient-server. Klient je označován manažer a server jako agent. Agenti jsou rozmístěni v na uzlech v síti. Na základě předem definovaných instrukcí ukládá získané parametry do databáze MIB. Veškerá komunikace mezi manažerem a agenty probíhá přes SNMP protokol pracujícím na aplikační vrstvě s využitím UDP protokolu. Použitím UDP protokolu SNMP disponuje velkou rychlostí, nevýhodou je však je možná ztráta zasílané informace. Verze 2 a vyšší již obsahuje kontrolu doručení a nemělo by tak dojít ke ztrátě informace. Existují celkem tři verze SNMP, první dvě verze používají pro autentizaci textové heslo, do třetí verze byla implementována autentizace pomocí jména, hesla a šifrování. Pro komunikaci je využíván port 161 na straně agenta pro dotazy a port 162 pro zasílání trapů na straně serveru. Trap lze popsat jako upozornění na určitou událost jako třeba překročení mezní hodnoty, výpadek služby, nebo porucha hardwaru. [17]

SNMP pracuje ve dvou režimech:

Agent přijímá dotazy od správce a zasílá mu zpět odpovědi, hodnoty můžou být odesílány více správcům a ti mohou zasílat dotazy kdykoliv.

Agent zasílá oznámení správci, pokud dojde k předem definované události. Může tak dojít při změně určité hodnoty nebo v pravidelných intervalech.

SNMP příkazy dostupné pro verzi 1 a 2 :

- get-request – získá data z MIB.
- get-next-request – získá data o objektu v MIB bez znalosti jeho OID, slouží postupnému procházení celým stromem.
- set-request – změní hodnotu proměnné agenta.
- get-response – odpověď agenta manažerovi po vykonání operace, kterou dostal příkazem.
- get-bulk – vyžádání si k přečtení skupinu informací z MIB, slouží pro urychlení komunikace.
- trap – příkaz odesílaný bez předchozí žádosti, od agenta manažerovi, při výskytu předem definované události. Nevýhodou je absence potvrzení o doručení, agent tak nemá jistotu, zda byla zpráva doručena. [12]

5.4.1 MIB

MIB je databáze, do které jsou ukládány informace využívané systémem správy. Každou událost v SNMP je možné v databázi MIB dohledat pomocí jednoznačného identifikátoru OID. OID je tvořen posloupností čísel oddělených tečkou. Vznikne tak že vezme hodnota nadřazeného prvku a na konec posloupnosti se přidá tečka a číslo aktuálního.

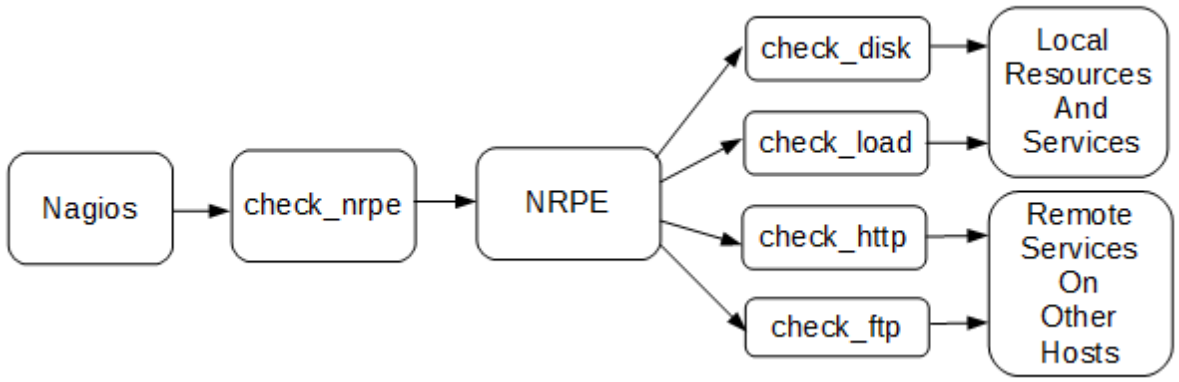
5.5 RRDTool

Round-Robin tool je open source nástroj sloužící ke zpracování a ukládání časově závislých dat, která mohou být reprezentována teplotou, vytížeností procesoru, síťovým provozem atd.

Data jsou ukládána do databáze round-robin s konstantní velikostí v čase. Tyto data je možné zobrazit v grafu, lze zobrazit různě velká časová období a také více proměnných. V grafu tedy můžeme zobrazit vytížení serveru a jeho paměti, nebo volné místo na discích. [13]

5.6 NRPE

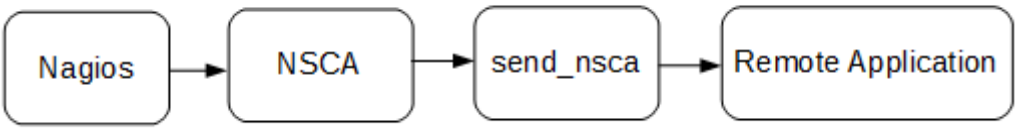
Nagios Remote Plugin Executor slouží pro testování služeb na vzdálených stanicích jako je zatížení procesoru, operační paměti, přihlašování uživatelů a mnoho dalších. Pro přenos dat mezi sledovanou stanicí a serverem je použitou SSL. NRPE lze je složeno ze dvou částí, démona a klienta. Klientská část programu nrpe_check se připojuje na NRPE hosta, kde provede daný test a výsledky pošle Nagios serveru. Samotná služba není naročná na výkon, pracuje pouze jako jeden proces tzv. démon, který spouští jednotlivé předem definované skripty. [14]



Obr. 13: Architektura NRPE

5.7 NSCA

Nagios Service Check Acceptor také slouží pro pasivní testování. Skládá se ze dvou částí, démona nsca a klientské části s názvem send_nsca. Pracuje obdobně jako SNMP trap, tj. na vzdálené stanici se vykoná potřebný test pomocí Nagios pluginů a klientská část send_nsca předá výsledek serverové části NSCA, která je zpracuje. Postup je tak opačný než u NRPE. [14]



Obr. 14: Architektura NSCA

5.8 Xinetd

Extended Internet daemon je open-source super démon pracující pod unixovým systémem. Velká spousta aplikací a služeb má vlastní démony naslouchající na určitém portu. Pro úsporu systémových prostředků se tak o velkou část démonů stará xinetd. Nahrazuje tak jednotlivé démony a poslouchá na daných portech a čeká na požadavek komunikace. Po přijetí žádosti o komunikaci ověří její oprávnění a spustí příslušného démona, který již danou službu zprostředkuje. Po skončení celého procesu se démon ukončí a naslouchá opět jen xinetd.

Velkou výhodou xinetd je centrální správa ostatních démonů, a tím i zvýšení bezpečnosti. K dalším přednostem patří možnost kontroly přístupu podle IP adresy, možnost omezení přístupu podle denní doby, ochrana služeb před útoky DoS, nebo přeměrování požadavku na jinou stanici. [15]

5.9 Praktická realizace monitoringu pomocí nástroje Zabbix

5.9.1 Instalace systému Zabbix

Před instalací zabbixu, je nutné doinstalovat služby a knihovny potřebné pro činnost serveru.

```
yum install httpd httpd-devel
yum install mysql mysql-server
yum install php php-cli php-common php-devel php-pear php-gd
php-mbstring php-mysql php-xml
```

Spustíme webový server a databázový systém.

```
service httpd start
service mysqld start
mysql_secure_installation
```

Připojíme repozitář obsahující zabbix a spustíme instalaci.

```
rpm –
uvhhttp://repo.zabbix.com/zabbix/2.2/rhel/5/x86_64/zabbix-
release-2.2-1.el5.noa

yum install zabbix-server-mysql zabbix-web-mysql zabbix-agent
zabbix-java-gateway
```

Zabbix vytvořil konfigurační soubor /etc/httpd/conf.d/zabbix.conf, ve kterém je nutné změnit časové pásmo na Europe/Prague. Nyní spustíme webový server.

```
service httpd restart
```

Založení databáze v MySQL

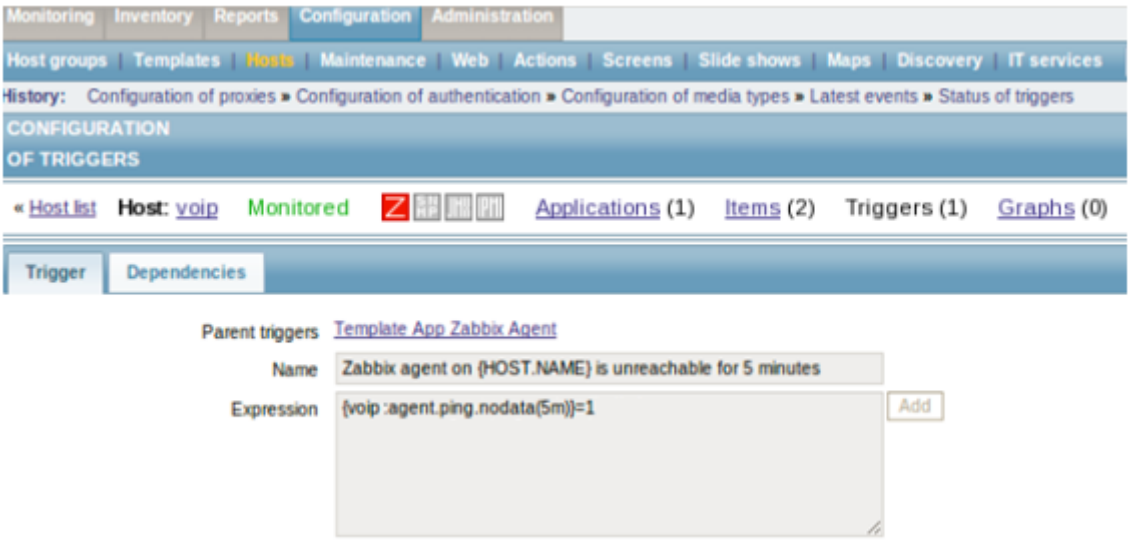
```
mysql -u root -p

mysql> CREATE DATABASE zabbix CHARACTER SET UTF8;

mysql> GRANT ALL PRIVILEGES on zabbix.* to 'zabbix'@'localhost'
IDENTIFIED BY 'SECRET_PASSWORD';

mysql> FLUSH PRIVILEGES;

mysql> quit

mysql -u zabbix -p zabbix < /usr/share/doc/zabbix-server-mysql-
2.2.2/create/schema.sql

mysql -u zabbix -p zabbix < /usr/share/doc/zabbix-server-mysql-
2.2.2/create/images.sql

mysql -u zabbix -p zabbix < /usr/share/doc/zabbix-server-mysql-
2.2.2/create/data.sql
```

Po úspěšném vytvoření databáze spustíme server a přejdem do webového prohlížeče, kde dokončíme instalaci.

```
service zabbix-server start
```

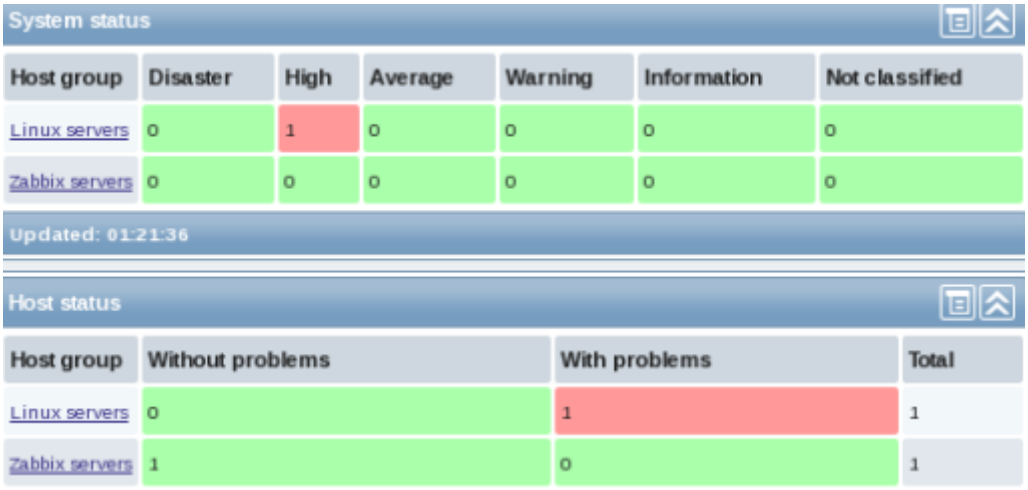
5.9.2 Otestování notifikace e-mailem ze systému Zabbix

Na obrázku 15 máme detail triggeru vytvořeného pro hosta s názvem voip. Tento trigger slouží k testování dostupnosti hosta. Pomoci funkce ping je kontrolován server každých pět minut.



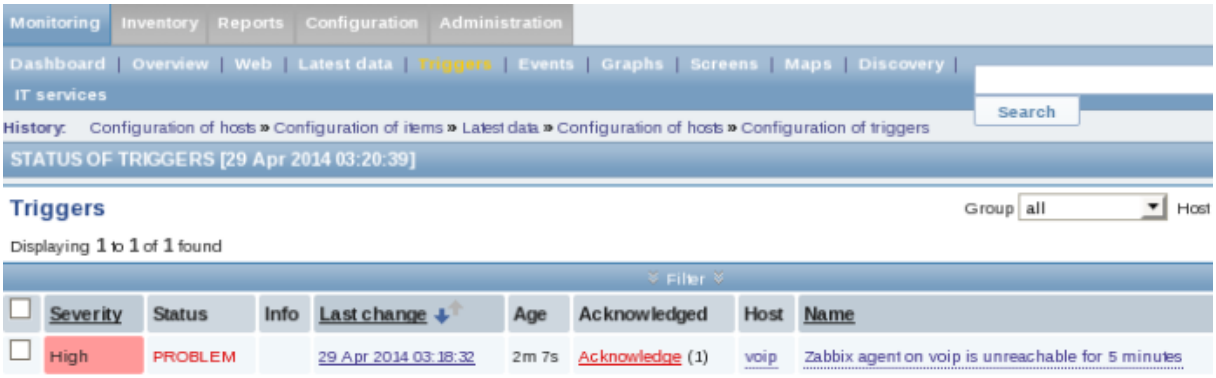
Obr. 15: Detail Triggeru

Obrázek 16 znázorňuje stavy hostů a Zabbix serveru. Pro otestování odesílání notifikací byl odpojen námi monitorovaný VoIP server.



Obr. 16: Nedostupnost VoIP serveru

Po rozkliknutí záložky Triggers, můžeme vidět na obrázku 17, o který trigger se jedná. Jde o hosta voip s vysokou závažností problému. Konkrétně se jedná o nedostupnost serveru po dobu pěti minut.



Obr. 17: Trigger nedostupnosti po 5 minutách

Na Obrázek 18 došlo k odeslání e-mailu správci serveru na adresu ondroid7@gmail.com. Dále lze vidět strukturu e-mailu, kde předmětem je oznámení o problému. Samotná zpráva pak obsahuje název triggeru, status a závažnost.

Obr. 18: Odeslání e-mailu správci

5.10 Praktická realizace monitoringu pomocí nástroje Nagios

5.10.1 Instalace Nagios Serveru

Stáhneme Nagios a Nagios plugins pomocí příkazu wget.

```
wget
http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-
4.0.1.tar.gz
```

```
wget https://www.nagios-plugins.org/download/nagios-plugins-
2.0.tar.gz
```

Rozbalíme stažené tarbally.

```
tar -xvf nagios-4.0.1.tar.gz
```

```
tar –xvf nagios-plugins-2.0.tar.gz
```

V adresáři Nagiosu spustíme konfigurační soubor.

```
./configure --with-command-group=nagcmd
```

Provedeme kompilaci a nainstalujeme.

```
make all
make install
```

Dokončíme instalaci.

```
make install-init
make install-commandmode
make install-config
```

Po úspěšné instalaci jádra Nagiosu je nutné nakonfigurovat webové rozhraní.

```
make install-webconf
```

V tomto kroku vytvoříme heslo pro přihlášení nagios admina.

```
htpasswd -s -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

Restartujeme Apache.

```
service httpd start
```

V předposledním kroku přejdeme do adresáře nagios plugins a provedeme jejich instalaci.

```
cd /root/nagios/nagios-plugins-1.5 ./configure --with-nagios-
user=nagios --with-nagios-group=nagios
make
make install
```

Restartujeme Nagios.

```
service nagios start
```

Nagios je nyní připraven k použití. Webový interface je dosupný na http://Your-server-IP-address/nagios. Zde je nutné vyplnit přihlašovací údaje nagiosadmin a námi zvolené heslo.

5.10.2 Instalace hosta na monitorované stanici

Jako první aktualizujeme potřebné knihovny a doinstalujeme chybějící.

```
yum install -y gcc glibc glibc-common gd gd-devel make net-snmp
openssl-devel
```

Vytvoříme nový účet s názvem Nagios a nastavíme heslo.

```
useradd nagios
passwd nagios
```

Vytvoříme adresář pro stažení pluginů, do něj stáhneme aktuální Nagios Plugins pomocí příkazu wget a rozbalíme.

```
wget https://www.nagios-plugins.org/download/nagios-plugins-
2.0.tar.gz
tar –xvf nagios-plugins-2.0.tar.gz
```

Přejdeme do adresáře a pomocí následujích příkazů provedeme instalaci.

```
cd nagios-plugins-2.0
./configure
make
make install
```

Nastavíme práva na adresář pluginu.

```
chown nagios.nagios /usr/local/nagios
chown -R nagios.nagios /usr/local/nagios/libexec
```

Ověříme si, zda je nainstalovaný Xinetd démon, pokud ne tak doinstalujem.

```
yum install xinetd
```

Stáhneme NRPE plugin příkazem wget. Rozbalíme, zkompilujeme a nainstalujeme.

```
wget http://garr.dl.sourceforge.net/project/nagios/nrpe-
2.x/nrpe-2.15/nrpe-2.15.tar.gz
tar xzf nrpe-2.15.tar.gz
./configure
make all
```

Nainstalujeme NRPE plugin démon pod službou xinetd.

```
make install-plugin
make install-daemon
make install-daemon-config
make install-xinetd
```

Nyní otevřeme soubor /etc/xinetd.d/nrpe a přidáme localhost a IP adresu Nagios Monitoring Serveru. V souboru /etc/services přidáme na konec souboru řádku s portem 5666, na kterém běží NRPE démon a restartujeme xinetd službu.

Přidáme pravidlo pro povolení provozu NRPE démona ve firewallu a uložíme iptables, aby pravidlo bylo obsaženo i po restartu systému.

```
iptables -A INPUT -p tcp -m tcp --dport 5666 -j ACCEPT
service iptables save
```

5.10.3 Instalace hosta na straně serveru

Přesuneme se do adresáře nagios, stáhneme NRPE plugin pomocí příkazu wget a rozbalíme.

```
wget http://garr.dl.sourceforge.net/project/nagios/nrpe-
2.x/nrpe-2.15/nrpe-2.15.tar.gz
tar xzf nrpe-2.15.tar.gz
```

Zkompilujeme a nainstalujeme NRPE.

```
./configure
make all
make install-daemon
```

Pro přidání hosta je nutné vytvořit dva konfigurační soubory hosts.cfg a services.cfg v adresáři /usr/local/nagios/etc/. Editujeme soubor s názvem nagios.cfg, ve kterém přidáme cestu k námi vytvořeným konfiguračním souborům. Soubory hosts.cfg a services.cfg slouží k definování hosta a monitorovaných služeb.

Definici NRPE příkazu přidáme do commands.cfg.

```
define command{
command_name check_nrpe
command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
}
```

Ověříme Nagios konfigurační soubor zda neobsahuje errory a restartujeme Nagios.

```
/usr/local/nagios/bin/nagios –v/usr/local/nagios/etc/nagios.cfg
service nagios restart
```

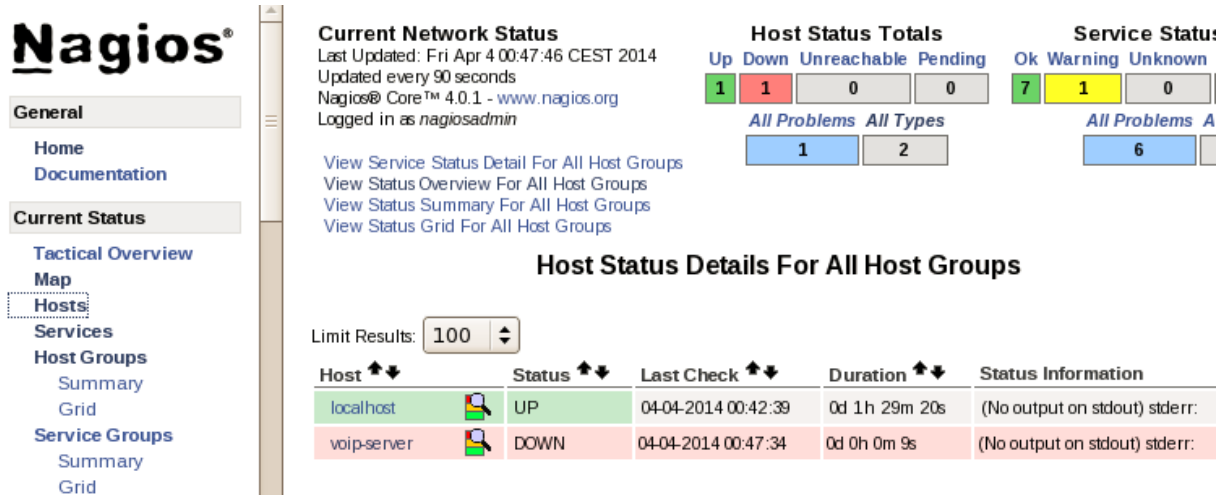
5.10.4 Otestování notifikace e-mailem ze systému Nagios

Na obrázku 19 máme náhled konfiguračního souboru VoIP serveru. Kde můžeme vyčíst, že bude monitorována dostupnost po celý den, každý den v týdnu v intervalu tří minut.

```
## Default Linux Host Template
define host{
name                            linux-box
use                             generic-host
check_period                    24x7
check_interval                  3
retry_interval                  1
max_check_attempts              5
check_command                   check-host-alive
notification_period             24x7
notification_interval           3
notification_options            d,r
contact_groups                  admins
register                        0
}

## Default
define host{
use                             linux-box
host_name                       voip-server
alias                           voipS
address                         10.0.0.5
host
}
```

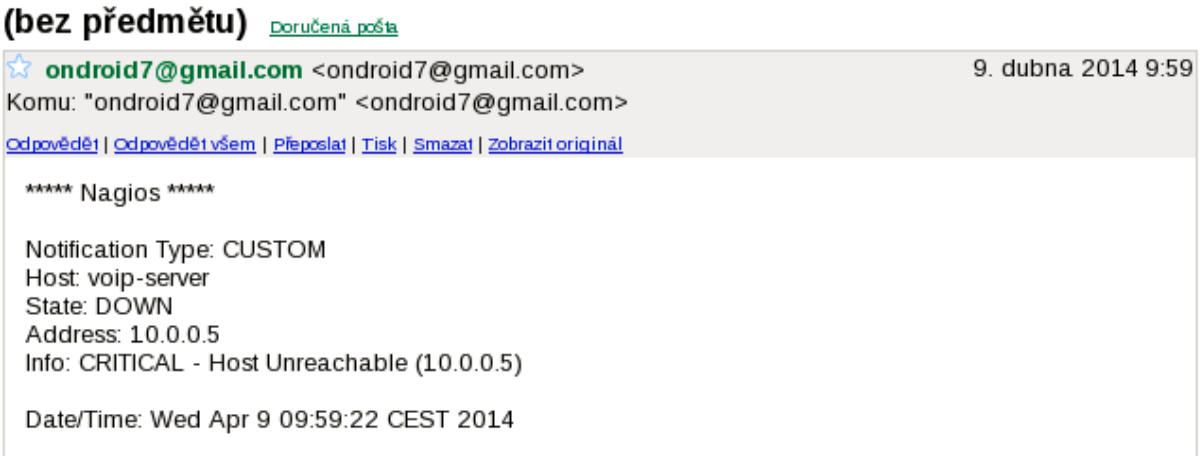
Obr. 19: Náhled konfiguračního souboru host.cfg

Na obrázku 19 došlo k odpojení hosta s názvem voip-server. Úkolem je otestování zda při výpadku serveru či po uspěšném útoku a následné nedostupnosti dojde k odeslání notifikace.



Obr. 19:Nedostupnost VoIP serveru

Obrázek 20 ukazuje strukturu e-mailu odeslaného systémem Nagios správci serveru. Lze z něj vyčíst, že došlo k nedostupnosti hosta voip-server s adresou 10.0.0.5 a čas poslední provedené kontroly.



Obr. 20:Notifikace e-mailem

5.11 Zhodnocení monitorovacích systémů

V části věnované monitoringu, bylo mým úkolem sledovat VoIP server a jeho dostupnost. Pokud dojde k odpojení stanice a bude nedostupná určitý čas, musí být její správce neprodleně informován na e-mail. Tato fuknce byla otestována na obou systémech. Na systému zabbix byl server kontrolován každých pět minut a monitoring Nagiosem v intervalu tří minut. Na obrázcích lze vidět, že obě služby fungují spolehlivě.

Porovnání obou systémů:

Konfigurační soubory systému Nagios jsou uloženy v jednoduchém textovém formátu. Umožňují tak jednoduchou úpravu, přidávání nových akcí, nebo hromadnou modifikaci pro více stanic najednou, ať již ručně, nebo pomocí skriptů.

Zatímco zabbix pro konfiguraci používá databázi, kterou nelze přímo modifikovat. Všechny změny tak musí být prováděny přes GUI nebo přes XML importy.

Data jsou u Nagiosu sbírána za pomoci agenta běžícího na monitorované stanici dvěma způsoby. Agent automaticky zasílá aktualizace na server nebo čeká na příchozí dotazy ze serveru a zasílá požadovanou odpověď.

Zabbix pracuje prostřednictvím agenta běžícího na každém monitorovaném subjektu. Informace jsou shromažďovány a odesílány na server v pravidelných intervalech.

Nagios ukládá historii alertů a kontrol jednotlivých služeb, z nich lze poté vytvořit reporty pro jednotlive skupiny. V základní instalaci není tvorba grafů, ale lze doinstalovat pomocí doplňků. Nýbrž zabbix má rozsáhlé reporty a tvorbu grafů již v základu.

Výchozí GUI Nagiosu obsahuje monitoring hostů, upozornění na alerty a prohlížení reportů. Nelze však přidávat, ani jinak upravovat hosty. Všechny uprávy lze provádět jen v konfiguračních souborech, nebo za pomoci externích nástrojů. Ty je však nutné doinstalovat. Nagios lze také plně ovládat pomocí příkazové řádky, což je velkým plusem pro vzdálený monitoring.

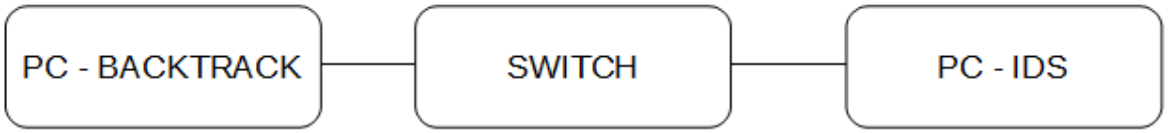
Zabbix má primární rozhraní přes GUI, to však omezuje například úpravy přes příkazovou řádku.

Systém Zabbix je více monolitický s menší možností úprav, s kvalitním grafickým prostředím, nevýhodou však je složitější instalace a počáteční nastavení služeb. Na rozdíl od Nagiosu, který je velmi flexibilní, modulární a lehce modifikovatelný a vyniká snadnou instalací a úpravou konfigurací. Rychlejší a jednodušší cestou k požadovanému monitoringu VoIP serveru byl tak Nagios.

6 Nasazení bezpečností opatření na základě provedených teoretických rozborů

6.1 Topologie pro testování IDS systémů

Následující kapitola popisuje jednotlivé kroky pro nasazení IDS systémů v praxi. Zde budou popsány příkazy pro systém Linux, nastavení konfiguračních souborů a topologie sítě. Na obrázku 21 je schéma zapojení sítě, ve které testování probíhalo. Obsahuje dvě počítačové stanice a switch. Na prvním počítači byl instalován systém BackTrack 5 R3, ze kterého byly generovány útoky invite flood. Spojení s druhým počítačem probíhalo pomocí switche. Na počítači druhém byl instalován systém Ubuntu ve verzi 12.04, na kterém byly postupně testovány jednotlivé IDS systémy.



Obr. 21: Topologie pro testování nástroje Snort, Suricata a Bro

6.2 Generování útoku

Na obrázku 22 je zobrazeno generování útoku invite flood, při kterém ze stanice s adresou 10.0.0.1 je zasláno 100 000 paketů na stanici s adresou 10.0.0.2 s portem 5060.

```
root@bt:/pentest/voip/inviteflood# ./inviteflood eth0 201 10.0.0.2 10.0.0.2 100000

inviteflood - Version 2.0
              June 09, 2006

source IPv4 addr:port   = 10.0.0.1:9
dest   IPv4 addr:port   = 10.0.0.2:5060
targeted UA             = 201@10.0.0.2

Flooding destination with 100000 packets
sent: 100000
```

Obr. 22: Generování útoku invite flood

6.3 Instalace a použití IDS Snort

Jako první si stáhneme knihovnu daq-1.1.1 a nainstalujeme.

```
sudo tar zxvf daq-1.1.1.tar.gz
cd daq-1.1.1
```

```
sudo ./configure
sudo make
sudo make install
```

Poté pokračujeme instalací knihovny libret-1.12.

```
sudo tar zxvf libdnet-1.12.tgz
cd libdnet-1.12/
sudo ./configure
sudo make
sudo make install
sudo ln -s /usr/local/lib/libdnet.1.0.1 /usr/lib/libdnet.1
```

Následuje samotná instalace Snortu.

```
sudo tar zxvf snort-2.9.4.5.tar.gz
cd snort-2.9.4.5
sudo ./configure --prefix=/usr/local/snort --enable-sourcefire
sudo make
sudo make install
sudo mkdir /var/log/snort
sudo mkdir /var/snort
sudo groupadd snort
sudo useradd -g snort snort
sudo chown snort:snort /var/log/snort
```

V souboru snort.conf, který je umístěn v adresáři /usr/local/snort/etc/ zadáme cestu k námi vytvořenému pravidlu sip.rules.

Na obrázku 23 můžeme vidět, žebylo načteno celkem deset pravidel sloužících k odhalení útoků ve VoIP telefonii. Nás bude konkrétně zajímat funkčnost pravidla pro odhalení útoku invite flood.

```
++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
WARNING: /usr/local/snort/rules/sip.rules threshold (in rule) is deprecated;
use detection_filter instead.

10 snort rules read
    10 detection rules
    0 decoder rules
    0 preprocessor rules
10 Option Chains linked into 3 Chain Headers
0 Dynamic rules
++++++++++++++++++++++++++++++++++++++++++++++++++
```

Obr. 23: Načtení pravidel

Snort spustíme pomocí následujícího příkazu a budeme generovat útok invite flood z druhého počítače, na kterém byl instalován systém BackTrack.

```
sudo snort -u snort -g snort \ -c
/usr/local/snort/etc/snort.conf -i eth0
```

6.4 Reakce na útok – Snort

Z logového souboru na obrázku 24 můžeme vidět, že byl odhalen útok podle pravidla Rule for alerting of INVITE flood attack. Definice tohoto pravidla zní, že pokud z jakékoliv IP adresy přijde na port 5060 více jak 100 paketů během 60 sekund, bude tato událost klasifikována jako útok.

```
#Rule for alerting of INVITE flood attack:

alert ip any any -> any 5060 (msg:"COMMUNITY SIP INVITE message
flooding"; content:"INVITE"; depth:6; threshold: type both,
track by_src, count 100, seconds 60; classtype:attempted-dos;
sid:100000158; rev:2;)
```

```
[**] [1:100000158:2] COMMUNITY SIP INVITE message flooding [**]
[Classification: Attempted Denial of Service] [Priority: 2]
04/26-11:08:13.649272 00:0C:29:8B:80:76 -> 00:0C:29:C2:DC:31 type:0x800 len:0x443
10.0.0.1:9 -> 10.0.0.2:5060 UDP TTL:64 TOS:0x0 ID:36218 IpLen:20 DgmLen:1077
Len: 1049
```

Obr. 24: Odhalení útoku invite flood nástrojem Snort

6.5 Instalace a použití IDS Suricata

Před začátkem instalace samotné Suricaty je nutné aktualizovat a nainstalovat chybějící knihovny nutné pro korektní běh systému.

```
sudo apt-get -y install libpcre3 libpcre3-dbg libpcre3-dev \
build-essential autoconf automake libtool libpcap-dev libnet1-dev \
libyaml-0-2 libyaml-dev zlib1g zlib1g-dev libcap-ng-dev libcap-ng0 \
make libmagic-dev
```

Následně stáhneme aktuální verzi Suricaty a rozbalíme do složky suricata-1.4.1

```
wget http://www.openinfosecfoundation.org/download/suricata-1.4.1.tar.gz
tar -xvzf suricata-1.4.1.tar.gz
cd suricata-1.4.1
```

Poté provedeme samotnou instalaci Suricaty.

```
./configure --prefix=/usr --sysconfdir=/etc --localstatedir=/var
make
sudo make install
sudo ldconfig
```

Po dokončení instalace pomocí příkazu vytvoříme složky potřebné pro chod systému, aktualizujeme knihovnu s pravidly a provedeme základní nastavení konfiguračního souboru suricata.yaml.A spustíme následujícím příkazem.

```
./configure && make && make install-full
sudo suricata -c /etc/suricata/suricata.yaml -i eth0
```

6.6 Reakce na útok – Suricata

Suricata má rozdílné ukládání do logů, než je u Snortu. Na rozdíl od Snortu, který ukládá všechny události do jednoho souboru, Suricata ukládá události do více souborů, podle druhu útoku. Ale hlavní záznam se nachází ve var/log/suricata/fast.log.

Podle obrázku 25 můžeme vidět, že došlo k odhalení útoku Suricatou podle stejného pravidla Rule for alerting of INVITE flood attack, jako u Snortu.

```
#Rule for alerting of INVITE flood attack:

alert ip any any -> any 5060 (msg:"COMMUNITY SIP INVITE message
flooding"; content:"INVITE"; depth:6; threshold: type both,
track by_src, count 100, seconds 60; classtype:attempted-dos;
sid:100000158; rev:2;)
```

```
04/22/2013-05:19:07.427276 [**] [1:100000158:2] COMMUNITY SIP INVITE message flooding [**]
[Classification: Attempted Denial of Service] [Priority: 2] {UDP} 10.0.0.1:9 -> 10.0.0.2:5060
```

Obr. 25: odhalení útoku invite flood nástrojem Suricata

6.7 Instalace a použití IDS Bro

Před instalací samotného IDS opět provedeme aktualizaci nutných knihoven a balíčků pro chod IDS Bro.

```
apt-get install libncurses5-dev g++ bison flex libmagic-dev
libgeoip-dev libssl-dev build-essential python-dev libpcap-dev
cmake swig2.0 libssl0.9.8
```

Stáhneme si aktuální verzi Bro, rozbalíme a nainstalujeme.

```
wget http://www.bro-ids.org/downloads/release/bro-2.0.tar.gz

tarzxvf bro-2.0.tar.gz

cdbro-2.0

./configure--prefix=/opt/bro2

make&&makeinstall
```

Před samotným spuštěním IDS provedeme základní konfiguraci těchto tří souborů. Ujistíme se, že v souboru network.cfg máme nastavené správné rozhraní, které je naším cílovým

pro monitoring. A v posledním zmiňovaném máme možnost nastavit e-mailovou adresu a interval v jakém budeme chtít zasílat logovací soubory.

```
/opt/bro2/etc/node.cfg          // konfigurace rozhraní.
/opt/bro2/etc/networks.cfg      // nastavení vnitřní sítě
/opt/bro2/etc/broctl.cfg        // nastavení odesílání výstrah
```

Nyní můžeme spustit program broctl, který najdeme v /usr/local/bro

```
./broctl
```

Příkazem install načteme duležité soubory pro chod systému a příkazem start spustíme samotnou instanci Bro.

```
[BroControl] > install
[BroControl] > start
```

Pro odhalení útoku již nebude využito pravidel Snortu, ale Bro skriptů, které jsou uloženy v adresáři /$PREFIX/share/bro/policy.

6.8 Reakce na útok – Bro

Ukládání údálostí se liší oproti IDS Snort a Suricata ve struktuře. Záznamy jsou děleny do souborů, podle druhu útoku. Tím rozumíme, že pokud je veden útok na port 80, je záznam uložen do logu http.log. Pro tetstování bylo využito vlastních pravidel programu Bro. Bro na rozdíl od předchozích dvou IDS při odhalení hrozby neupozorní uživatele, plní spíše monitorovací funkci.

Na obrázku 26 lze vidět výpis ze souboru Notice.log, na kterém je odhalení útoku programem Bro. Upozorněno bylo jen na větší počet přijatých paketů, malá část byla zahozena a chybí IP adresa útočníka.

```
1366636378.508312 - - - - - - PacketFilter::Dropped_Packets 56 packets
dropped after filtering, 95469 received, 95469 on link - - - - - bro
Notice::ACTION_LOG 6 3600.000000
```

Obr. 26: odhalení útoku invite flood nástrojem Bro

6.9 Zhodnocení IDS

Trh disponuje velkým výběrem IDS systémů, jak komerčních, tak volně dostupných. Všechny tři testované systémy jsou open-source a jsou volně ke stažení. Co se týče ovládání, může být Bro naročnější pro začínajícího uživatele systému UNIX. Samotní vývojáři Bro popisují jako experimentální IDS, tudíž by mělo být spíše použito pro testování a experimentování v síti, než jako hlavní IDS. Popřípadě může být jako doplněk hlavního IDS. Ve srovnání se Suricatou a Snortem je Bro účinnější pro vysokorychlostní sítě. Z hlediska signatur je Bro více sofistikovanější v porovnaní s ostatními. Díky propracovanému jazyku, ve kterém sou psány, mohou být signatury mnohem detailněji specifikovány. Samotné Bro již v základu dokáže odhalit známé útoky, k přidání dalších funkcí a signatur je nutné znát základy Bro jazyka.

Velkou výhodou Snortu a Suricaty je snadná instalace a uvedení do provozu, kdežto Bro je naročnější v nasazení a celkovému porozumění systému. Další velké plus pro první dva jmenované je možnost instalace grafického rozhraní, které pro Bro není dostupné a je tak nutností pro uživatele znalost příkazů unixového shellu. Snort i Suricata jsou oba dostupné pro většinu dnes rozšířených systémů včetně Windows, zatímco Bro lze instalovat pouze na UNIX a Mac OS X.

O prvenství Snortu mezi IDS systémy má také velkou zásluhu jeho tým tvořící aktualizace, dokumentace a nové pravidla. Je komerčně podporován společností SourceFire a velkou uživatelskou základnou přispívající také k jeho rozvoji.

7 Závěr

Cílem této bakalářské práce bylo otestovat tři IDS nástroje a realizovat monitoring VoIP serveru pomocí dvou systémů. Vstupem do problematiky je seznámení s principy zabezpečení a rozbor použitých technologií použitých v praktické části. V první polovině praktické části byl řešen monitoring pomocí dvou velmi rozšířených systémů, Nagios a Zabbix, kde hlavním úkolem bylo monitorovat činnost VoIP serveru a v případě výpadku služby informovat správce.

Druhá polovina praktické části je zaměřena na testování IDS systémů instalovaných na VoIP server proti útoku invite flood. Kapitola obsahuje postupy instalací jednotlivých IDS systémů, ukázku použitého pravidla, generování a zachycení útoku. Následuje porovnání systémů.

Na závěr bych chtěl podotknout, že je velmi nutné mít zřetel na bezpečnost a bezproblémový chod VoIP serveru, který je velmi důležitým prvkem VoIP infrastruktury. Praktickou částí bylo ověřeno, že bezpečnost lze zvýšit použitím některého z IDS a nepřetržitým monitoringem služby.

Realizací této práce jsem získal cenné zkušenosti v oblasti bezpečnosti a monitoringu serverů. A věřím, že tyto nově nabyté vědomosti budou důležitým faktorem při výběru mého budoucího zaměstnání.

Použitá literatura

[1] ENDORF, Carl. *Detekce a prevence počítačového útoku*. 1. vyd. Praha: Grada, 2005. ISBN 80-247-1035-8.

[2] BEALE, Jay, Andrew R BAKER a Joel ESLER. *Practical VoIP Security: IDS and IPS toolkit*. Rockland: Syngress Pub, 1975, xxxiv, 730 p. ISBN 978-008-0489-551.

[3] DOSTÁLEK, Libor a Marta VOHNOUTOVÁ. *Velký průvodce infrastrukturou PKI a technologií elektronického podpisu*. vyd. 1. Brno: Computer Press, 2006, 534 s. ISBN 80-251-0828-7.

[4] PORTER, Thomas. *Building Secure Servers With Linux (2003)*. Rockland: Syngress Publishing, 2006, 563 s. ISBN 15-974-9060-1.

[5] DUC Václav, LESNÍK David. Stavový firewall na linuxu. [Online] [cit. 2014-02-03.]http://wh.cs.vsb.cz/sps/images/3/34/Stavovy_firewall_na_linuxu.pdf.

[6] www.voip-info.org [online]. [cit. 2014-02-03].

http://www.voip-info.org/wiki/view/Asterisk+firewall+rules

[7] GRANDISON, Tyrone a Evimaria TERZI. *Intrusion Detection Technology* [online]. IBM Almaden Research Center650 Harry Road, San Jose, CA 95120, 2007 [cit. 2014-05-03].

[8] BEALE, Jay, Andrew R BAKER a Joel ESLER. *Snort: IDS and IPS toolkit*. Burlington, MA: Syngress, c2007, xxxiv, 730 p. ISBN 978-159-7490-993.

[9] SOMMER, Robin**.** International Computer Science Institute. [Online] [cit. 2014-02-03.] http://www.icir.org/robin/slides/cybersummit-bro.pdf.

[10] VORLÍČEK, Jaroslav. LINUXALT. *Suricata IDS/IPS* [online]. [cit. 2014-02-03]. http://lvb.sti.fce.vutbr.cz/public/LinuxAlt_2010/2010_11_07_LA_07_Suricata/2010_11_07_LA_07_Suricata.pdf.

[11] KOCJAN, Wojciech. *Learning Nagios 3.0: a detailed tutorial to setting up, configuring, and managing this easy and effective system monitoring software*. Birmingham, U.K.: Packt Pub., c2008, v, 301 p. From technologies to solutions. ISBN 978-1-84719-518-0.

[12] BIGELOW, Stephen J. *Mistrovství v počítačových sítích: správa, konfigurace, diagnostika a řešení problémů*. Vyd. 1. Překlad Petr Matějů. Brno: Computer Press, 2004, 990 s. ISBN 80-251-0178-9.

[13] RRDTool. In*: Wikipedia: the free encyclopedia* [online]. *San Francisco (CA): Wikimedia Foundation, 2001-* [cit. 2014-02-03]. *cs.wikipedia.org/wiki/RRDTool*

[14] Nagios Addons. [online]. [cit. 2014-05-04].

http://nagios.sourceforge.net/docs/3_0/addons.html

[15] TURNBULL, James. *Hardening Linux*. New York: Distributed to the Book trade in the United States by Springer-Verlag, c2005, xxvii, 552 p. ISBN 15-905-9444-4.

[16] PORTER, Thomas. *Practical VoIP security*. Syngress Publishing, 2006, 563 s. ISBN 15-974-9060-1.

[17] PORTER, Thomas. *Building Secure Servers With Linux (2003)*. Rockland: Syngress Publishing, 2006, 563 s. ISBN 15-974-9060-1.